

# STUDIENHEFT

**MMDE12C**

**Entwicklung von  
Webanwendungen**

**Serverseitige Programmierung**



---

**Das Studienheft und seine Teile sind urheberrechtlich geschützt. Jede Nutzung in anderen als den gesetzlich zugelassenen Fällen ist nicht erlaubt und bedarf der vorherigen schriftlichen Zustimmung des Rechteinhabers. Dies gilt insbesondere für das öffentliche Zugänglichmachen via Internet, Vervielfältigungen und Weitergabe. Zulässig ist das Speichern (und Ausdrucken) des Studienheftes für persönliche Zwecke.**

---

**MMDE12C**

# **Entwicklung von Webanwendungen**

## **Serverseitige Programmierung**

**Torsten Schreiber**

---

Werden Personenbezeichnungen aus Gründen der besseren Lesbarkeit nur in der männlichen oder weiblichen Form verwendet, so schließt dies das jeweils andere Geschlecht mit ein.

Falls wir in unseren Studienheften auf Seiten im Internet verweisen, haben wir diese nach sorgfältigen Erwägungen ausgewählt. Auf die zukünftige Gestaltung und den Inhalt der Seiten haben wir jedoch keinen Einfluss. Wir distanzieren uns daher ausdrücklich von diesen Seiten, soweit darin rechtswidrige, insbesondere jugendgefährdende oder verfassungsfeindliche Inhalte zutage treten sollten.

# Entwicklung von Webanwendungen

## Serverseitige Programmierung

### Inhaltsverzeichnis

<b>Einleitung</b>	1
<b>1 Grundlagen datenbankgestützter Webanwendungen</b>	3
1.1 Was ist eine Webanwendung?	3
1.1.1 Statische und dynamische Webseiten	5
1.1.2 Rich Internet Application (RIA)	7
1.2 Webprogrammierung	8
1.2.1 Clientseitige Programmierung	8
1.2.2 Serverseitige Webprogrammierung	10
1.2.3 Einsatzgebiete	12
1.2.4 Sicherheit	13
Zusammenfassung	14
<b>2 Einführung in serverseitige Programmierung mit PHP</b>	16
2.1 Einführung in PHP	16
2.2 Historie	16
2.3 Vorteile	17
2.4 Funktionsweise	18
2.5 Erste Schritte mit PHP	19
2.5.1 Rechnen	21
2.5.2 PHPinfo	25
2.5.3 PHP-Code mehrfach verwenden	27
2.5.4 Formular auswerten	30
Zusammenfassung	34
<b>3 MySQL</b>	35
3.1 Vorteile von MySQL	37
3.2 Zusammenarbeit zwischen PHP und MySQL	37
3.3 Ablauf	38
Zusammenfassung	42
<b>4 Editoren, IDEs und Frameworks</b>	44
4.1 Editoren	44
4.2 Integrierte Entwicklungsumgebung – IDE	47
4.3 PHP-Frameworks	52
Zusammenfassung	56

<b>5 XML</b>	57
5.1 Was ist XML?	57
5.2 Aufbau eines XML-Dokuments	57
5.2.1 Prolog	58
5.2.2 XML-Elementbaum	59
5.3 XML-Dateien und PHP	61
Zusammenfassung	65
<b>Schlussbetrachtung</b>	66
<b>Anhang</b>	
A. Lösungen der Aufgaben zur Selbstüberprüfung	67
B. Glossar	69
C. Literaturverzeichnis	73
D. Abbildungsverzeichnis	74
E. Quellcodeverzeichnis	75
F. Sachwortverzeichnis	76
G. Einsendeaufgabe	79

## Einleitung

Herzlich willkommen zum Studienheft „Entwicklung von Webanwendungen – Serverseitige Programmierung“.

Dieses Studienheft spricht viele Themengebiete im Zusammenhang mit serverseitiger Programmierung kurz an, stellt Grundlagen und Konzepte vor und bietet so einen fundierten theoretischen Einstieg in das Thema.

Umfangreiche und komplexe Webanwendungen sind heutzutage allgegenwärtig und für uns selbstverständlich. Kaum ein Unternehmen, eine Institution oder Organisation verzichtet darauf, sich im World Wide Web ausgiebig zu präsentieren. Diese Entwicklung ist nicht neu. Doch dort, wo zu Beginn des WWW noch das Angebot von statischen Inhalten stand, trat spätestens mit der Entwicklung zum Web 2.0 vor etwa 10 Jahren der Wunsch nach einem Mehrwert in Form von Interaktion, Individualität und Leistungsfähigkeit hinzu. Die statische Homepage wurde mehr und mehr abgelöst durch dynamische Webanwendungen.

Technologisch basieren viele dieser Anwendungen auf der Kombination von PHP als serverseitige Skriptsprache und MySQL als Datenbankmanagementsystem. Die Popularität dieser Kombination ist bis heute ungebrochen und wird weltweit am häufigsten für die Entwicklung von Webanwendungen eingesetzt.

### Lerninhalte und Lernziele

In diesem Studienheft erarbeiten wir uns einige Grundlagen zur serverseitigen Programmierung im Rahmen von Webprogrammierung. Wir sehen uns an, was PHP ist und wie es funktioniert.

Moderne Webanwendungen verarbeiten häufig große Mengen an Daten. Diese stammen in der Regel aus Datenbanken. Wir werden deshalb in diesem Heft zur serverseitigen Programmierung auch einen Blick auf das relationale Datenbankmanagementsystem MySQL werfen. Neben allgemeinen Informationen zu MySQL betrachten wir an einem Beispiel auch die Zusammenarbeit zwischen PHP und einer MySQL-Datenbank.

Erste Schritte mit PHP lassen sich ohne komplizierte Infrastruktur mit einem einfachen Texteditor gehen. Warum das in umfangreichen Projekten anders gemacht werden sollte, sehen wir uns in der Lektion „Editoren, IDEs und Frameworks“ an.

Die letzte Lektion dieses Studienhefts gibt Ihnen in einer Art Exkurs einen kurzen Überblick über XML, den Aufbau von XML-Dokumenten und einen Ansatz für die Arbeit mit PHP und XML-Dokumenten.

Im Einzelnen lernen Sie in diesem Studienheft,

- was eine Webanwendung ist,
- was den Unterschied zwischen statischen und dynamischen Websites ausmacht,
- was eine RIA ausgemacht hat,
- wie die Eigenschaften und Einsatzgebiete clientseitiger sowie serverseitiger Programmierung aussehen,
- welche Geschichte, welche Vorteile und welche Funktionsweise PHP besitzt,
- wie PHP grundsätzlich funktioniert,

- was MySQL ist,
- wie PHP und MySQL zusammenarbeiten,
- was der Unterschied zwischen Editoren und IDEs ist,
- wo die Vorteile bei der Nutzung integrierter Entwicklungsumgebungen liegen,
- was Frameworks sind und wie Sie ein geeignetes PHP-Framework auswählen,
- was unter XML zu verstehen ist,
- wie ein XML-Dokument grundsätzlich aufgebaut ist und
- wie Sie lesend mit PHP auf eine XML-Datei zugreifen.

Torsten Schreiber



# 1 Grundlagen datenbankgestützter Webanwendungen

*In dieser Lektion erarbeiten wir uns einige Grundlagen im Zusammenhang mit datenbankgestützten Webanwendungen. Sie erfahren, was eine Webanwendung ist. Außerdem lernen Sie die Unterschiede zwischen statischen und dynamischen Websites sowie einige Grundbegriffe zur Webprogrammierung kennen. Zum Schluss schauen wir uns noch kurz an, welche Vorteile Webanwendungen haben und welchen Herausforderungen sich Entwickler von Webanwendungen stellen müssen.*

Beginnen wir mit der Beantwortung der Frage, was eine Webanwendung eigentlich ist.

## 1.1 Was ist eine Webanwendung?

Webanwendungen, häufig auch Webapplikationen genannt, sind weiterhin in aller Munde. Die meisten Anwender von Webanwendungen werden auf die Frage, was aus ihrer Sicht eine Webanwendung ist, vermutlich sagen: Google, eBay, Amazon oder Facebook. Und damit liegen sie insoweit richtig, dass die vorgenannten Unternehmen allesamt berühmte Websites mit umfangreichen Webanwendungen betreiben. Die für die Entwicklung und den Betrieb dieser Webanwendungen eingesetzten Techniken unterscheiden sich im Detail durchaus voneinander. Die grundlegenden Konzepte und eingesetzten Technologien sind für den Großteil der Webanwendungen aber ähnlich.

Eine erste Begriffsdefinition für Webanwendung, die wir im Laufe dieser Lektion noch etwas erweitern werden, lautet:

Eine Webanwendung oder Webapplikation ist eine Anwendung, die auf einem Webserver ausgeführt wird.

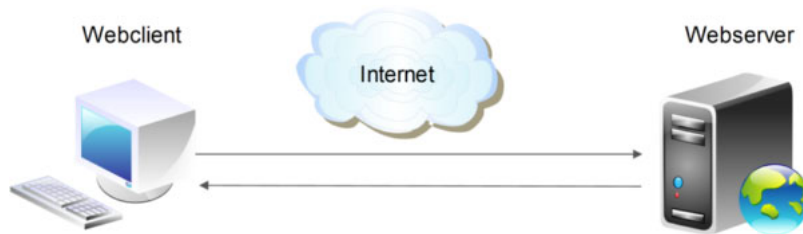


Als Webclient dient in der Regel der Webbrowser, beispielsweise der Edge, Internet Explorer, Firefox, Chrome oder Opera. Nur in seltenen Fällen werden eigene Webclients entwickelt.

Die Datenübertragung findet auf Basis von TCP/IP (Transmission Control Protocol/Internet Protocol) – auf Anwendungsebene durch das Protokoll HTTP (Hypertext Transfer Protocol) – statt.

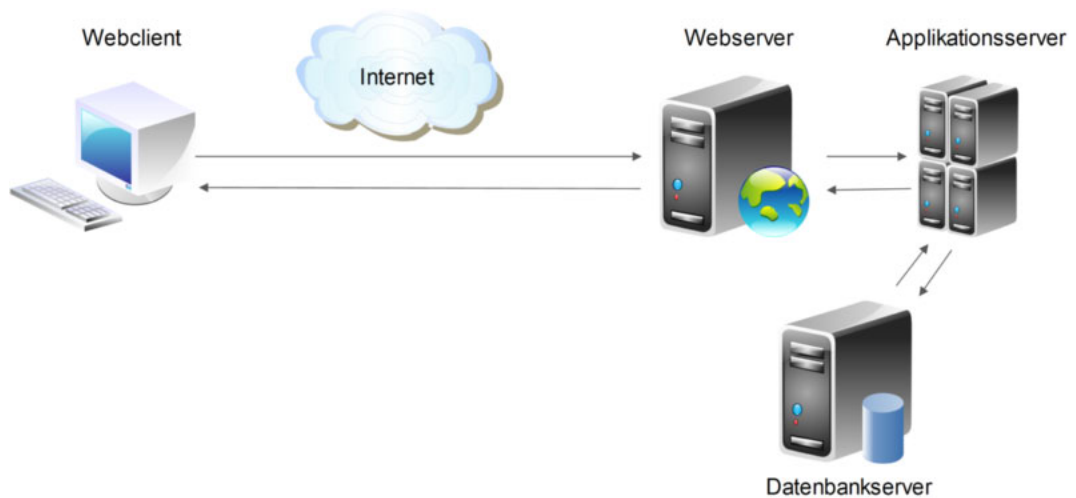
TCP/IP ist eine Protokollfamilie, die die Grundlage der Kommunikation im Internet bildet. Ein Protokoll, auch Netzwerkprotokoll oder Netzprotokoll genannt, enthält vereinfacht gesagt Regeln, die den Austausch von Daten zwischen Computern und/oder Diensten ermöglichen.

Die räumliche Entfernung zwischen dem Webclient und dem Webserver ist für eine Webanwendung ohne Bedeutung, denn beide sind über das Internet miteinander verbunden.



**Abb. 1.1:** Infrastruktur für einfache Webanwendungen

In umfangreichen, professionellen Umgebungen werden Webanwendungen nicht mehr direkt auf dem Webserver ausgeführt, sondern auf zusätzlichen Applikationsservern. Die Anwendungsdaten werden über Datenbankmanagementsysteme zur Verfügung gestellt.



**Abb. 1.2:** 3-Schicht-Architektur für umfangreiche Webanwendungen

Der Ablauf ist dabei folgendermaßen:

- Der Client stellt eine Anfrage an den Webserver.
- Die Webapplikation wird entweder auf dem Webserver ausgeführt oder die Anfrage wird an einen Applikationsserver weitergeleitet und die Anwendung wird dort ausgeführt.
- Der Austausch erforderlicher Daten erfolgt über den Datenbankserver, bevor auf umgekehrtem Weg das Ergebnis zurück an den Client gesendet wird.

In Anlehnung an die Anzahl der Beteiligten spricht man in diesem Zusammenhang von einer 3-Schicht-Architektur (engl. Three-Tier-Architecture):

1. Präsentationsschicht (Webbrowser)
2. Applikationslogik (Web- u. Applikationsserver)
3. Datenbankschicht (Datenbankserver)

**Hinweis:**

Wenn wir von Webclient, Webserver oder Datenbankserver sprechen, meinen wir in diesem Heft immer die Software und nicht die Hardware. Ein Webserver kann zwar auf einem eigenen Rechner laufen, muss das aber nicht zwangsläufig. Es ist genauso möglich, dass der Webserver und der Datenbankserver auf derselben physischen Hardware ausgeführt werden. Genauso ist es möglich, dass viele Webserver auf der einen und ein oder mehrere Datenbankserver auf einer anderen Maschine ausgeführt werden.

Abhängig ist das vor allem von den Anforderungen in Bezug auf Leistungsfähigkeit und Sicherheit, die an das gesamte System gestellt werden.

**1.1.1 Statische und dynamische Webseiten**

Im Zusammenhang mit der Ausführung von Webapplikationen werden Technologien eingesetzt, die bereits zu Beginn des World Wide Web (WWW) Ende der 1980er- und Anfang der 1990er-Jahre Verwendung fanden:

- ein Webserver,
- ein Webclient und
- die für die Kommunikation zwischen beiden erforderlichen Protokolle.

Der Webserver ist eine Serverapplikation, die auf HTTP-Anfragen reagiert und entsprechende Antworten erzeugt. Der Webclient stellt Anfragen an den Webserver und zeigt dessen Antworten an.

Bei klassischen Webseiten wurden unter Zuhilfenahme der Auszeichnungssprache HTML (Hypertext Markup Language) statische Texte, Grafiken und Hyperlinks in Webseiten auf Webservern gespeichert. Solche Webseiten erlaubten nur das Betrachten dieser Inhalte und die Verzweigung über ebenfalls statische Hyperlinks. Interaktion erfolgte beispielsweise über HTML-Formulare.

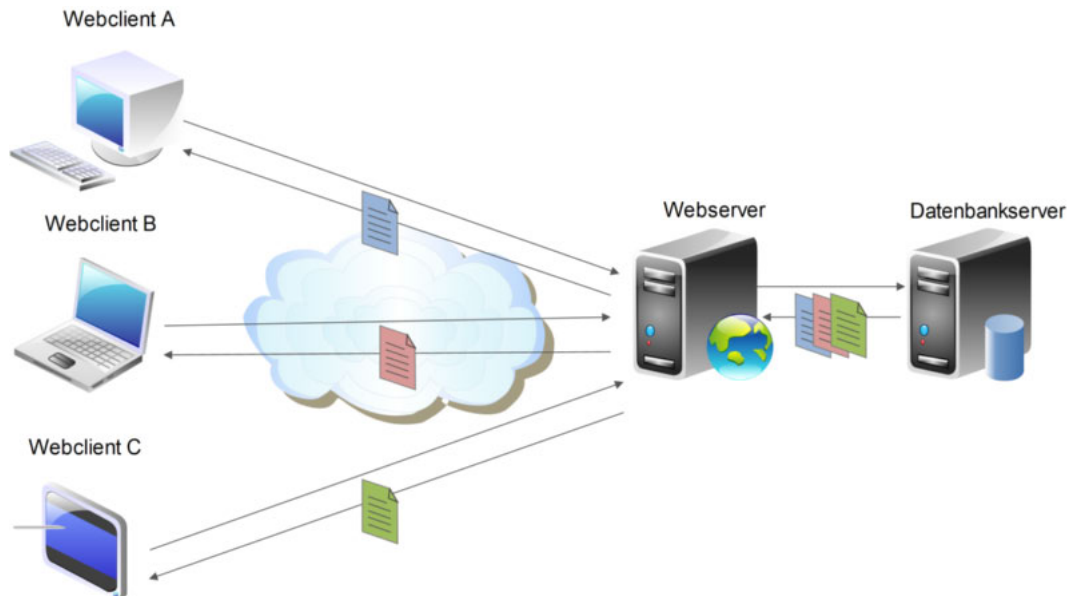


**Abb. 1.3:** Abruf statischer Webseiten

Der Webclient richtet eine Anfrage (Request) nach einem bestimmten Dokument an den Webserver. Im besten Fall findet der Webserver das Dokument am angegebenen Ort und liefert es als Antwort (Response) an den Client zurück.

Zunehmend wuchs allerdings der Wunsch nach zusätzlichen Möglichkeiten der Individualisierung und der Interaktion. Benutzer wollten und sollten Daten in Abhängigkeit von ihrem persönlichen Profil ansehen, bearbeiten oder hinzufügen können.

Webapplikationen bieten genau das! Die Webseiten liegen dann nicht mehr statisch auf dem Webserver, sondern werden in Abhängigkeit von bestimmten Parametern **dynamisch erzeugt**. Außerdem ermöglichen Webanwendungen die Interaktion zwischen dem Benutzer und den Anwendungsdaten auf dem Server.



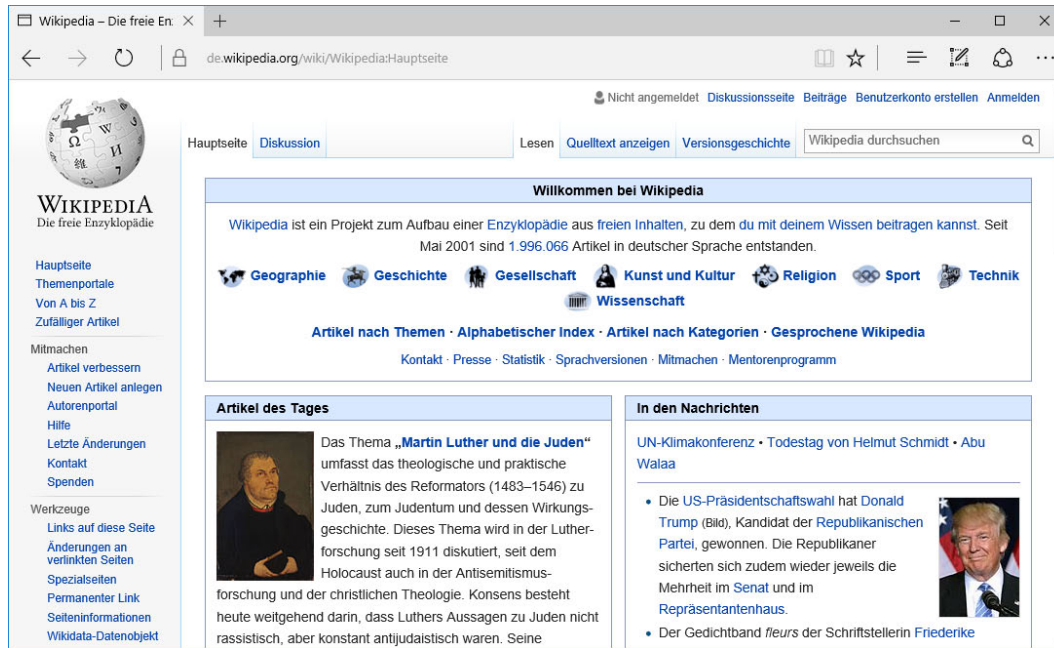
**Abb. 1.4:** Dynamisch erzeugte Webseiten

Wie diese Dynamik erreicht werden kann, wird uns im Folgenden noch grundsätzlich beschäftigen. Eines deutet sich bereits an: Ohne Programmierung wird das nicht funktionieren.

Ergänzen wir zum Schluss dieses Abschnitts noch kurz unsere Begriffsdefinition für „Webanwendung“, die wir im ersten Abschnitt begonnen hatten, um die zuletzt gewonnenen Erkenntnisse:



Eine Webanwendung ist eine Softwareanwendung, die auf einem Web- oder Applikationsserver im Internet bzw. Intranet ausgeführt wird, in der Regel über eine Datenbankbindung verfügt (datenbankgestützt), Webseiten dynamisch zur Laufzeit erzeugt und dem Benutzer die Interaktion mit der Anwendung und/oder den Anwendungsdaten ermöglicht.



**Abb. 1.5:** Wikipedia als Beispiel für eine datenbankgestützte Webanwendung

### 1.1.2 Rich Internet Application (RIA)

Webanwendungen, die in ihren Interaktionsmöglichkeiten und oft auch im gesamten Look & Feel wie Desktopanwendungen erschienen, wurden Rich Internet Applications (RIAs) genannt.

Sie boten beispielsweise Drag & Drop, komplexe Menüs oder waren grafisch besonders anspruchsvoll. Alles Eigenschaften, die von einer „normalen“ Browseranwendung zuvor nicht bekannt waren.

„RIA“ war eine Zeit lang ein Schlagwort, das sehr oft im Kontext mit der Entwicklung des World Wide Web zum Web 2.0 gefallen ist. Mittlerweile ist der Begriff etwas aus der Mode gekommen, weil das, was eine Rich Internet Application ursprünglich ausmachte und zu diesem Zeitpunkt neu und besonders war, mittlerweile zum Standard professioneller Webanwendungen wurde.

Wenn solche Begriffe zunehmend verschwinden, gibt es in der Regel zwei Ursachen. Entweder sie wurden bedeutungslos oder alltäglich. Für RIA gilt, dass die Techniken in der Entwicklung von Webapplikationen alltäglich wurden und so keine besondere Bezeichnung mehr notwendig war.

RIAs verwenden in der Regel dieselbe klassische Internettechnologie wie andere Webapplikationen auch. Besonders serverseitig gibt es häufig kaum Unterschiede.

Auf Clientseite kann das anders aussehen: Es gibt RIAs, die aus einem Browser gestartet werden, aber separat von diesem auf Betriebssystemebene laufen (browserunabhängige RIAs), und RIAs, die in einem speziellen Browser-Plug-in ausgeführt werden (Plug-in-basierte RIAs).

Die dritte und größte Gruppe der Rich Internet Applications verwendet keine zusätzlichen Hilfsmittel, sondern nutzt die Möglichkeiten, die ihnen moderne Webbrowser zur Verfügung stellen.

Diese Browser-RIAs verwenden häufig eine „Technologie“ namens **Ajax** (Asynchronous JavaScript And XML) und werden deshalb auch als Ajax-Anwendungen bezeichnet. Das Wort Technologie ist deshalb in Anführungszeichen eingeschlossen, weil es sich im engeren Sinne überhaupt nicht um eine Technologie handelt, sondern verschiedene Techniken unter einem Namen zusammengefasst werden.

Etwas mehr dazu und einen groben Einblick in die Funktionsweise von Ajax erhalten Sie im Abschnitt 1.2.1 „Clientseitige Programmierung“.

## 1.2 Webprogrammierung

Es gibt viele verschiedene Techniken, um Webseiten interaktiv und dynamisch zu gestalten. Nicht jede dieser Techniken macht aus einer Webseite eine Webanwendung. Manchmal dient die Technik auch nur dazu, bestimmte optische Effekte zu erzeugen.



Animation ist im Gegensatz zu Interaktion kein Kriterium einer Webanwendung.

In der Webprogrammierung wird zwischen **clientseitiger** und **serverseitiger Programmierung** unterschieden. Bei der Entwicklung von Webanwendungen werden beide Ansätze regelmäßig miteinander kombiniert.

### 1.2.1 Clientseitige Programmierung

Wie die Bezeichnung es vermuten lässt, versteht man unter clientseitiger Programmierung Skripte und Programme, die auf dem Client, das heißt, im Webbrowser ausgeführt werden. Haupteinsatzgebiet der clientseitigen Programmierung ist die unmittelbare Interaktion mit dem Anwender, zum Beispiel die Reaktion auf das Mausverhalten oder die sofortige Überprüfung von Eingabedaten in einem Formular vor der Übertragung zu einem Server.

Wichtige Techniken in diesem Bereich sind:

- **HTML** und **CSS** für die Darstellung,
- **JavaScript** als wichtigste Skriptsprache und
- **Ajax** für desktopähnliches Verhalten.

HTML und CSS sind Ihnen bereits sehr gut bekannt, sodass wir darauf nicht weiter eingehen müssen.

JavaScript ist die wichtigste Skriptsprache im Zusammenhang mit clientseitiger Programmierung. Den Namen erhielt JavaScript, da es syntaktische Ähnlichkeit zur Programmiersprache **Java** hat.

JavaScript stammt von dem Unternehmen Netscape und diente zunächst dazu, den Funktionsumfang des Netscape Navigators, eines Webbrowsers der frühen Generationen, zu erweitern. Zwischenzeitlich hatte JavaScript einen schlechten Ruf. Es galt als

unsicher und wurde deshalb in Unternehmen browserseitig häufig abgeschaltet. Mittlerweile wurde JavaScript diesbezüglich verbessert und findet auch in Unternehmen seither deutlich größere Akzeptanz.

JavaScript hat sich als clientseitige Skriptsprache absolut etabliert und findet in unzähligen Webapplikationen Anwendung, um vornehmlich<sup>1</sup> clientseitig bestimmte Funktionalität zur Verfügung zu stellen.

Ajax ist – wie zuvor bereits angedeutet – keine neue, eigenständige Technologie, sondern vereint mehrere in der Webprogrammierung bekannte Technologien unter einem gemeinsamen Namen und in einem gemeinsamen Ansatz. Dazu zählen vor allem:

- HTML
- JavaScript
- das Document Object Model (DOM)
- XMLHttpRequest

Besonders spannend an Ajax ist der asynchrone Datenaustausch im Hintergrund. Was bedeutet das?

Von klassischen Webanwendungen sind Sie in etwa ein Verhalten wie in folgendem Beispiel gewöhnt:

Nehmen wir an, Sie möchten über eine Suchmaschine etwas suchen. Sie geben die Internetadresse ein oder klicken auf einen entsprechenden Hyperlink. Sie warten einen Moment, bis die Startseite der Suchmaschine angezeigt wird. Anschließend geben Sie den Suchbegriff ein und klicken auf eine Schaltfläche „Suchen“ oder Sie drücken die Eingabetaste. Wieder warten Sie und als Ergebnis erhalten Sie vom Server eine neue Seite mit den Suchergebnissen. Diese Form der Datenübertragung wird als synchrone Datenübertragung bezeichnet. Benutzeraktivität und Verarbeitung auf dem Server wechseln sich ab.

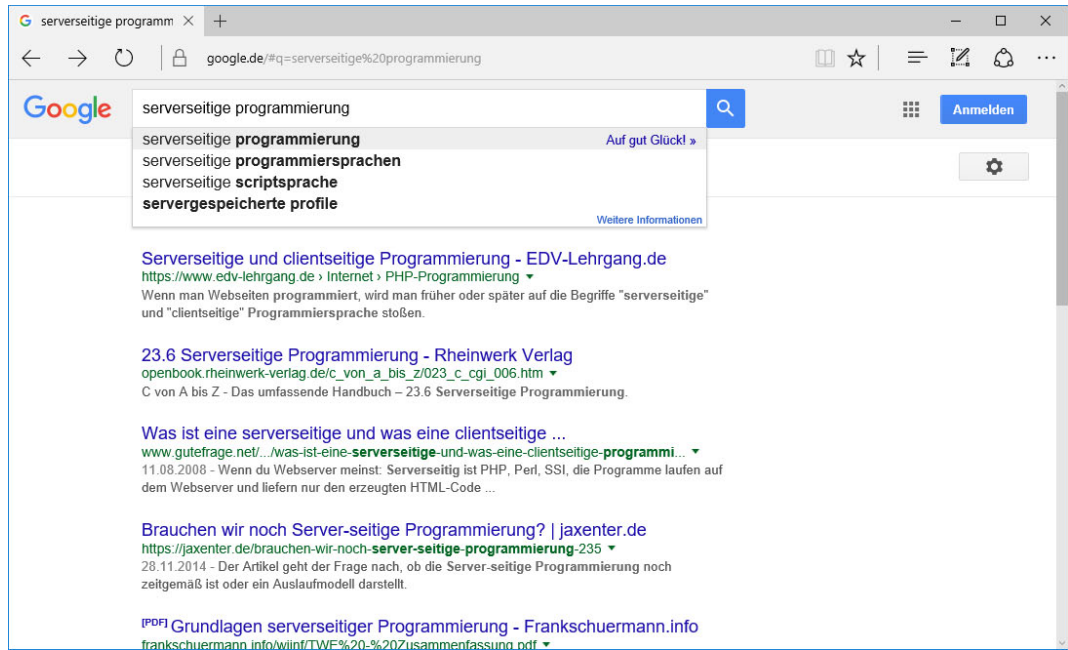
Asynchrone Datenübertragung ermöglicht dagegen, dass der Benutzer weiterarbeiten kann, während im Hintergrund Daten ausgetauscht werden. Hierbei wird, anders als bei der klassischen Variante, nicht die gesamte Seite neu geladen, sondern es werden nur die relevanten Teile einer Seite ausgetauscht.

Google verwendet beispielsweise in der Suche Ajax. Wir können unser Suchmaschinenbeispiel von oben hier als Ajax-Variante ansehen: Der Start bleibt derselbe, die Startseite erscheint. Bereits nach der Eingabe des ersten Zeichens in das Suchfeld werden Unterschiede deutlich. Sobald Sie einen Eintrag aus den Vorschlägen auswählen oder Ihr eingegebener Suchbegriff eine sinnvolle Auswertung durch Google zulässt, werden bereits erste Suchergebnisse angezeigt. Wohlgemerkt, ohne dass die Eingabetaste gedrückt oder die Suche über eine Suchen-Schaltfläche abgeschickt wurde.

---

1. Seit einiger Zeit kann JavaScript auch serverseitig eingesetzt werden, vgl. <https://de.wikipedia.org/wiki/Node.js>. Wir werden in diesem Studienheft aber nicht näher darauf eingehen.





**Abb. 1.6:** Suche mit Ajax-Technologie

Der Datenaustausch mit dem Server erfolgt über ein Objekt der Klasse XMLHttpRequest, einer Programmierschnittstelle, die es erlaubt, mit JavaScript HTTP-Anforderungen (Requests) an einen Server zu senden, ohne dass die Seite dabei neu geladen werden muss. Während der Browser auf die Antwort des Servers wartet, kann der Benutzer mit der Webanwendung weiterarbeiten. Nach der Verarbeitung sendet der Server die Antwort. Im Browser werden nun mithilfe von JavaScript und dem Document Object Model (DOM) nur die Bereiche aktualisiert, die durch die Antwort betroffen sind. Während der gesamten Zeit konnte der Anwender mit der Anwendung weiterarbeiten.

## 1.2.2 Serverseitige Webprogrammierung

Den Anwendungsgebieten serverseitiger Webprogrammierung sind wenig Grenzen gesetzt – egal ob Online-Shop, Suchmaschine, Content-Management-System oder soziales Netzwerk. Allen komplexen Webanwendungen ist gemeinsam, dass sie nicht mehr ohne die Anbindung an eine Datenbank auskommen. Die Anbindung an die Datenbank, die Verarbeitung und Aufbereitung der Daten wird beispielsweise mithilfe von serverseitiger Programmierung erledigt.

Mittlerweile gibt es mehrere Möglichkeiten, serverseitige Webanwendungen zu programmieren, und zahlreiche Schnittstellen, mit denen dynamische Webseiten auf dem Webserver generiert und gesendet werden. Grundsätzlich bestehen zwei Ansätze: Der eine verwendet fertige, kompilierte Programme, der andere verwendet Skriptsprachen.

Kompilierte Programme liegen auf dem Webserver und müssen dort nur noch gestartet werden. Skripte dagegen werden auf dem Webserver von einem Interpreter zur Laufzeit eingelesen, analysiert und ausgeführt.



Verschaffen wir uns kurz einen allgemeinen Überblick über gängige Techniken:

Der älteste und bis heute verbreitete Ansatz für serverseitige Dynamik im Web ist CGI<sup>2</sup>. CGI ermöglicht das Ausführen von Programmen auf dem Webserver. Dazu spezifiziert CGI die erforderlichen Schnittstellen für die Eingabe und Weiterleitung von Informationen vom Webclient zum Programm und für die Rückgabe zum Client. CGI ist keine Programmiersprache, sondern eine Protokollvereinbarung. CGI-Programme können mit vielen Programmiersprachen erstellt werden – zum Beispiel mit Perl oder C.

CGI belastet allerdings den Server sehr, da für jeden CGI-Aufruf ein neuer Prozess mit eigener Hauptspeicherreservierung aufgerufen wird. Unter ungünstigen Umständen führt das zu hoher Serverlast und langen Reaktionszeiten.

FastCGI (engl. fast = schnell) und SCGI (S für engl. simple = einfach) sind Erweiterungen zu CGI, die den Performancenachteil von CGI ausgleichen wollen. Beide nutzen Techniken, damit Anwendungen als Server-Modul ablaufen. Das verhindert das erneute Laden immer wiederkehrender Daten. Ein Overhead<sup>3</sup> wie bei CGI soll dadurch vermieden werden.

Servlets sind Java-Programme, die auf Webservern dynamisch Anfragen von Webclients beantworten können.

JSP (Java Server Pages) und ASP<sup>4</sup> (Active Server Pages) sind Technologien, um Code-Elemente in statisches HTML einzubetten. Der Code wird auf dem Webserver ausgeführt und erzeugt so dynamische Webseiten. JSP verwendet als Sprache Java. Für ASP stehen mehrere Skriptsprachen – zum Beispiel VB-Skript oder JScript – zur Verfügung.

Eine nicht unwesentliche Rolle in der serverseitigen Webentwicklung spielt heute auch ASP.NET (Active Server Pages .NET). ASP.NET ist eine auf dem .NET-Framework<sup>5</sup> basierende Technologie zum Entwickeln dynamischer Webseiten. Mit ASP.NET können Webanwendungen in allen Programmiersprachen entwickelt werden, die durch das .NET-Framework unterstützt werden – beispielsweise Visual Basic oder C#.

Das weltweit verbreitetste Instrument zur serverseitigen Webprogrammierung ist jedoch PHP (**PHP Hypertext Preprocessor**). Laut den Meinungsforschern von W<sup>3</sup>Techs<sup>6</sup> wurde PHP auf knapp über 80 Prozent aller untersuchten Seiten für die serverseitige Programmierung eingesetzt (Stand Januar 2017).

PHP ist eine extra für die Webprogrammierung entwickelte und nahezu exklusiv dort eingesetzte Skriptsprache. Die Sprache ist angelehnt an Perl und C, verfügt über sehr gute Möglichkeiten der Datenbankanbindung und findet häufig in der Kombination mit MySQL<sup>7</sup> Verwendung. PHP-Code kann direkt in HTML-Code eingebettet werden. Wenn der Client eine PHP-Datei anfordert, wird diese vom Webserver an den PHP-Interpreter übergeben, dort verarbeitet und das Ergebnis an den Webclient geschickt.

2. CGI = Common Gateway Interface (dt. allgemeine Datenaustausch-Schnittstelle)

3. Als Overhead werden in der IT Daten bezeichnet, die als Zusatzinformation zur Übermittlung oder Speicherung benötigt werden, aber keine Nutzdaten im engeren Sinne sind.

4. ASP wurde von Microsoft entwickelt und ist eine Art Vorreiter der serverseitigen Programmierung. Mit Erscheinen des .NET-Frameworks wurden Active Server Pages von ASP.NET abgelöst. Trotzdem werden auch heute noch kleinere Webauftritte mit ASP umgesetzt. Zunächst war ASP nur für IIS verfügbar, späterhin aber auch für weitere Webserver.

5. .NET ist eine Softwareplattform von Microsoft, u.a. bestehend aus einer Laufzeitumgebung und verschiedenen Klassenbibliotheken und Programmierschnittstellen.

6. [http://w3techs.com/technologies/overview/programming\\_language/all](http://w3techs.com/technologies/overview/programming_language/all)

Früher wurde der PHP-Interpreter übrigens mittels CGI ausgeführt; mit den oben genannten Nachteilen von CGI. Mittlerweile läuft er jedoch meistens als internes Webservermodul oder per FastCGI.

### 1.2.3 Einsatzgebiete

Die Einsatzgebiete von Webanwendungen sind vielfältig. Aufgrund der zuvor besprochenen Eigenschaften eignen sich Webanwendungen sowohl für klassische Unternehmensanwendungen, beispielsweise Artikel-, Kunden-, Mitarbeiterverwaltung oder Buchhaltungssysteme, wie für Internetanwendungen im engeren Sinn – zum Beispiel soziale Netzwerke, Shops, Suchmaschinen, Blogs, Foren, Web-Content-Management-Systeme usw.

### Vorteile

Durch die Verwendung des Webbrowsers als Client entstehen clientseitig keine zusätzlichen Kosten. Es muss keine Clientsoftware entwickelt oder beschafft und installiert werden. Außerdem entstehen im Zusammenhang mit der Webanwendung keine weiteren Wartungskosten beim Client.

Serverseitig fallen bei der Auswahl geeigneter Produkte keine bzw. nur geringe Lizenzkosten an. Die sehr weit verbreitete Kombination aus Linux-Betriebssystem, Apache Webserver, MySQL-Datenbankmanagementsystem und PHP als serverseitige Skriptsprache ist in den meisten Fällen lizenzkostenfrei erhältlich. Viele Hosting-Anbieter stellen eine solche Infrastruktur auch für geringes Entgelt zur Verfügung.

Für die Verbindung zwischen den Clients und dem Server dient das Internet. Es entstehen – außer gegebenenfalls Verbindungskosten – keine zusätzlichen Kosten und die Anwendung ist ortsunabhängig, das heißt mobil und von überall nutzbar.

Webapplikationen sind von ihrem Grundsatz her netzwerk- und multiuserfähig. Der Wartungsaufwand bei Aktualisierungen ist gering, da er nur serverseitig anfällt. Anwender arbeiten immer mit der neuesten Version der Anwendung.

Die Anwendungsdaten werden zentral gespeichert. Das erleichtert Datensicherungen und konsistente Datenhaltung.

### Herausforderungen

Die Tatsache, dass bei der Ausführung von Webapplikationen größtenteils „normale“ Internet-Technologie verwendet wird, bringt dem Entwickler nicht nur Vorteile, sondern stellt ihn auch vor besondere Herausforderungen. So muss sichergestellt werden, dass der parallele Zugriff mehrerer Benutzer auf die Anwendung und den Datenbestand reibungslos funktioniert. Die korrekte Darstellung im Browser muss ebenso gewährleistet werden wie der Umgang mit dem Browsercache und mit langsamen Internetverbindungen.

Eingangs haben wir festgestellt, dass als Übertragungsprotokoll für Webanwendungen, wie bei gewöhnlichen Webseiten auch, HTTP eingesetzt wird.

7. MySQL ist eines der weltweit führenden relationalen Datenbankmanagementsysteme (RDBMS) im Zusammenhang mit dynamischen Internetauftritten.

Der Umstand, dass HTTP ein zustandsloses<sup>8</sup> Netzwerkprotokoll ist, bei dem Werte direkt nach der Antwort des Servers verloren gehen, muss im Programmablauf in der Art besonders berücksichtigt werden, dass die Sitzungsdaten – zum Beispiel Werte von Variablen – entweder auf dem Client, auf dem Server oder auf beiden gespeichert und in geeigneter Weise weitergereicht werden.

Erreicht wird das beispielsweise durch die Verwendung von Cookies oder Sessions. Ein Cookie (dt. Keks, Plätzchen) ist eine Textdatei, die vom Server an den Client gesendet und dort gespeichert wird. Sobald die Website vom Client erneut aufgerufen wird, sendet der Client die im Cookie gespeicherten Informationen wieder zurück an den Server.

Als Session (dt. Sitzung) wird grundsätzlich eine Verbindung zwischen einem Client und einem Server bezeichnet. Bei Webanwendungen kann eine Session wegen der Verwendung von HTTP erst auf Anwendungsebene zustande kommen.

Beide Begriffe sollen in diesem einleitenden Grundlagenheft nur erwähnt werden.

### 1.2.4 Sicherheit

Die Sicherheit ist im Zusammenhang mit Webanwendungen von besonderer Bedeutung, sind diese doch über ein Netzwerk – und in der sehr großen Mehrheit sogar über ein weltweit öffentliches Netzwerk – zugänglich. Das Thema Sicherheit von Webanwendungen ist allerdings so umfangreich und komplex, dass wir uns in diesem grundlegenden Studienheft zu serverseitiger Programmierung darauf beschränken müssen, nur einige Sicherheitsaspekte zu nennen, um Ihnen einen sehr groben Überblick zu geben und Sie grundsätzlich für das Thema Sicherheit zu sensibilisieren.

Die Intention für einen Angriff auf Webanwendungen ist sehr unterschiedlich. Das geht von Vandalismus über monetäre beziehungsweise wirtschaftliche Vorteilmahme bis hin zum Platzen und Verteilen weiterer Schadsoftware. So werden über die Mailfunktionen gehackter Webanwendungen gerne in großen Mengen Spam und Schadsoftware versendet. In der Anwendung selbst ist davon häufig überhaupt nichts zu merken.

Die Interaktion zwischen Benutzer und Webanwendung erfolgt häufig über Formulareingaben. Ein erhebliches Sicherheitsrisiko für Ihre Anwendungen besteht dabei darin, dass auf diesem Weg schädlicher Programmcode in Ihre Anwendung eingeschleust wird. Es wird also Schadcode in Ihre Anwendung injiziert. Man spricht deshalb bei solchen Bedrohungen von Injections. Es gibt verschiedene Arten von Injections. Besonders häufig und auch besonders gefährlich sind dabei die sogenannten SQL-Injections. Dabei werden „unerwünschte“ SQL-Kommandos auf Ihre Datenbanken abgesetzt – beispielsweise, um Daten auszulesen oder zu manipulieren.

Zur Vermeidung von Injections ist die Trennung zwischen Eingabedaten und Befehlen zwingend. Je nach Art der Injection gibt es verschiedene Ansätze. Die zuvor erwähnten SQL-Injections lassen sich zum Beispiel sehr gut mit sogenannten **vorbereiteten Anweisungen** (prepared statements) verhindern.

8. Unter Zustandslosigkeit versteht man im Zusammenhang mit Netzwerkprotokollen die Eigenschaft, Anfragen komplett unabhängig voneinander zu behandeln. Das bedeutet, dass mehrere Anfragen desselben Clients oder Auftraggebers nicht in Bezug zueinander stehen.

Eine weitere Bedrohung besteht darin, dass Fehler in der Authentifizierung oder in der Sitzungsverwaltung ausgenutzt werden. Ziel des Angriffs ist es, fremde Zugangsdaten zu stehlen. Es wird mittelfristig zu Problemen führen, wenn Zugangsdaten nicht ordnungsgemäß gespeichert werden – zum Beispiel unverschlüsselte Passwörter – oder wenn der Zugriff auf Session-IDs (das sind eindeutige Identifizierer einer Sitzung) wegen deren öffentlicher Übertragung ermöglicht wird.

Ähnlich wie bei der Injection wird auch beim sogenannten Cross-Site-Scripting (XSS) Code in die Anwendung eingeschleust. Hier handelt es sich um Script-Code – zum Beispiel Java-Script –, um Eigenschaften des Browsers auszunutzen. XSS ist die häufigste Bedrohung für Webanwendungen überhaupt. Solche Probleme treten auf, wenn die Anwendung vom Benutzer eingegebene Daten übernimmt, ohne sie hinreichend zu prüfen.



Für Webanwendungen im Produktiveinsatz ist es also ganz besonders wichtig, dass Benutzereingaben niemals ungeprüft verarbeitet werden dürfen! Seien Sie hier besonders kritisch und vertrauen Sie keiner Eingabe. Prüfen Sie alle Benutzereingaben sehr sorgfältig.

Neben diesen zuvor genannten bedeutenden Bedrohungen für Webanwendungen gibt es noch viele, viele mehr. Sicherheit ist ein übergreifendes Thema, das Sie praktisch immer und auf allen Entwicklungsebenen beachten müssen.

#### Hinweis:

Für Hintergrundinformation zur Sicherheit von Webanwendungen empfehlen wir Ihnen auch die Lektüre der „BSI-Leitfäden zur Entwicklung sicherer Webanwendungen“<sup>9</sup>.

Sie richten sich an Auftraggeber, Projektmanager sowie Anwendungsentwickler, wurde im Auftrag des Bundesamtes für Sicherheit in der Informationstechnik (BSI) erstellt und werden durch dieses auch veröffentlicht.

## Zusammenfassung

Eine Webanwendung ist eine Softwareanwendung, die auf einem Web- oder Applikationsserver im Internet bzw. Intranet ausgeführt wird, in der Regel über eine Datenbank-anbindung verfügt (datenbankgestützt), Webseiten dynamisch zur Laufzeit erzeugt und dem Benutzer die Interaktion mit der Anwendung und/oder den Anwendungsdaten ermöglicht.

In der Webprogrammierung wird zwischen clientseitiger und serverseitiger Programmierung unterschieden. Bei der Entwicklung von Webanwendungen werden beide Ansätze regelmäßig miteinander kombiniert.

Für serverseitige Programmierung bestehen zwei Ansätze: kompilierte Programme oder Skriptsprachen.

9. [https://www.bsi.bund.de/DE/Publikationen/Studien/Webanwendungen/index\\_htm.html](https://www.bsi.bund.de/DE/Publikationen/Studien/Webanwendungen/index_htm.html)

Das Common Gateway Interface ist der älteste und bis heute verbreitete Ansatz für serverseitige Dynamik im Web. Weitere Techniken sind u.a.: FCGI, Servlets, ASP.NET und vor allem PHP.

Sicherheit ist ein hoch relevantes Thema für Webanwendungen. Gefahren sind zum Beispiel Injections und Cross-Site-Scripting.

### Aufgaben zur Selbstüberprüfung

Überprüfen Sie nun bitte Ihr neu erworbenes Wissen. Lösen Sie die Aufgaben zunächst selbstständig und vergleichen Sie anschließend Ihre Lösungen mit den Angaben im Anhang.

- 1.1 Was ist JavaScript und woher stammt der Name?
- 1.2 Nennen Sie die Schichten bei einer 3-Schicht-Architektur.
- 1.3 Nennen Sie einen gravierenden Nachteil bei der Verwendung von CGI in der serverseitigen Programmierung.

## 2 Einführung in serverseitige Programmierung mit PHP

*Nachdem Sie in der ersten Lektion grundlegende Konzepte und Technologien im Zusammenhang mit Webanwendungen kennengelernt haben, wenden wir uns jetzt PHP zu.*

*Im ersten Abschnitt erfahren Sie etwas über die Geschichte von PHP und über Eigenschaften der Sprache, die sie so populär gemacht haben. Außerdem stellen wir Ihnen im Detail vor, wie PHP funktioniert.*

*Anschließend erfahren Sie, wie PHP praktisch eingesetzt wird.*

Beginnen wir damit, wofür PHP steht und wie sich PHP historisch entwickelt hat.

### 2.1 Einführung in PHP

PHP steht für PHP Hypertext Preprocessor. PHP ermöglicht serverseitige Programmierung und damit auch die Entwicklung von Webapplikationen.

Zunächst wurde PHP vor allem eingesetzt, da es hervorragend die einfache Auswertung von HTML-Formularen unterstützt, mit denen ein Benutzer Daten an eine Website senden kann.

Selbstverständlich kann PHP aber sehr viel mehr.

### 2.2 Historie

PHP ist Nachfolger des 1995 von Rasmus Lerdorf entwickelten und zunächst nur auf Perl basierenden PHP/FI (Personal Home Page/Forms Interpreter). 1997 wurde PHP/FI durch die ersten Alphaversionen von PHP 3 abgelöst. PHP/FI lag zu der Zeit in der Version 2.0 vor und verfügte bereits über einige Funktionen, wie sie heute von PHP bekannt sind.

Auch wenn PHP 3 der offizielle Nachfolger von PHP/FI ist, wurde es von Andi Gutmans und Zeev Suraski 1997 komplett neu geschrieben. Aus PHP/FI wurde kurz PHP, was nun bereits für **PHP Hypertext Preprocessor** stand. Im Juni 1998 wurde die Version **PHP 3** offiziell freigegeben.

Im Jahr 2000 kam **PHP 4** mit komplett neuem Kern. Der Funktions- und Leistungsumfang stieg mit dieser Version gewaltig. Zahlreiche Webserver und Programmierschnittstellen wurden nun unterstützt. Mit dieser Version begann die ganz große Verbreitung von PHP in der Webentwicklung.

Nach mehreren Entwicklungsjahren erschien **PHP 5** im Jahre 2004 mit einem neuen Objektmodell und vielen zusätzlichen Funktionen. Seit Version 5 unterstützt PHP vollständig die objektorientierte Programmierung.

Am 3. Dezember 2015 erschien das Major-Release der derzeit aktuellen Version **PHP 7**.

**Hinweis:**

Die PHP-Version 6 hat es nur im Entwicklungsstadium gegeben. Da die dort verfolgten Ansätze nicht zu Ende entwickelt und wieder verworfen wurden, kam es bereits 2014 zu der Entscheidung, dass die nächste Hauptversion PHP 7 heißen soll. PHP 6 wurde damit übersprungen.

Am Anfang der Entwicklung von PHP wurde es von „richtigen“ Programmierern regelmäßig belächelt und nicht ernst genommen. Es galt unter anderem als chaotisch, was zwar auch an PHP selbst, aber vor allem an den Programmen und Skripten lag, die in den Webanwendungen Verwendung fanden. Die Vorbehalte wurden aber zunehmend geringer und die Anhängerschaft in gleichem Maße größer. Heutzutage verwenden viele der größten Internetauftritte serverseitig PHP und die bedeutendsten Web-Content-Management-Systeme sind mit PHP entwickelt worden.

## 2.3 Vorteile

Die ungebrochene Popularität von PHP basiert auf einigen Eigenschaften. Sehen wir uns die wichtigsten einmal an:

- Der Einstieg in PHP ist einfach. Bereits schnell sind erste Programmiererfolge zu erzielen.
- PHP ist leicht zu benutzen, denn PHP-Befehle können einfach in HTML eingebettet werden.
- PHP wurde über einen langen Zeitraum entwickelt und ist dabei gereift. Größere Kinderkrankheiten und Sicherheitsprobleme sind nicht mehr zu erwarten. Mit PHP 7 wurde auch die Performance erheblich verbessert.
- PHP ist auf die Programmierung von Webanwendungen spezialisiert.
- Mit PHP können Sie prozedural oder objektorientiert programmieren.
- PHP ist Open-Source-Software und damit frei verfügbar. Auch eine Entwicklungs- und Testumgebung steht kostenlos zur Verfügung.
- PHP kann auf allen gängigen Betriebssystemen verwendet werden.
- PHP unterstützt viele verschiedene Datenbanksysteme.
- Nahezu alle Webhosting-Angebote stellen mittlerweile PHP-Unterstützung bereit.

Ein zusätzlicher und nicht zu unterschätzender Vorteil ergibt sich aus der weiten Verbreitung von PHP sowie der großen und aktiven PHP-Community. Sollten Sie während Ihrer Programmertätigkeit auf Probleme stoßen, die Sie nicht direkt selbst lösen können, wird Ihnen in einem der vielen Foren zum Thema PHP bestimmt geholfen.

Zudem existieren bereits zu vielen Aufgabenstellungen fertige PHP-Skripte, die häufig auch zum Download angeboten werden. Sie müssen diese gegebenenfalls nur noch an Ihren konkreten Anwendungsfall anpassen.



Gerade für größere Projekte bietet sich dem schon versierten PHP-Programmierer eine Reihe von PHP-Frameworks an, die ihn bei der effizienten Entwicklung unterstützen können. Frameworks liefern eine Art Rahmen beziehungsweise ein Gerüst mit einer Vielzahl vorgefertigter Funktionalitäten, in dem das eigene Projekt entwickelt werden kann.

Das Angebot an PHP-Frameworks ist vielfältig. Die Auswahl eines geeigneten Frameworks ist nicht einfach und muss vor allem in Anlehnung an die Anforderungen des eigenen Projekts und des eigenen Programmierstils getroffen werden. Außerdem ist die Einarbeitung in ein Framework manchmal recht aufwendig, sodass auf eine entsprechend gute Dokumentation des Frameworks geachtet werden sollte.

Wir werden später in diesem Studienheft noch einmal darauf zurückkommen.

## 2.4 Funktionsweise

Sehen wir uns im Folgenden kurz an, wie PHP-Dateien ausgeführt werden.



**Abb. 2.1:** Funktionsweise von PHP

Im ersten Schritt sendet der Client, in der Regel ist das der Webbrowser, eine Anfrage an den Webserver. Beispielsweise, indem der URL in die Adresszeile eingegeben wird oder auf einen Hyperlink geklickt wird.

Der Webserver sucht und findet im zweiten Schritt die Datei im Dateisystem. Anhand der Dateinamenserweiterung identifiziert der Server die Datei als PHP-Datei und sendet sie im dritten Schritt zur weiteren Verarbeitung an den PHP-Interpreter, der auf dem Server ausgeführt wird.

Welche Dateinamenserweiterungen vom Webserver in welcher Form interpretiert werden sollen, wird in einer seiner Konfigurationsdateien festgelegt. Neben der üblichen und empfohlenen Endung **.php** treffen Sie bei Ihrer Arbeit hin und wieder vielleicht auch auf PHP-Dateien, die auf `.php3`, `.php4` oder `.php5` enden und anhand der Ziffer bereits nach außen verdeutlichen, zu welcher PHP-Version sie kompatibel sind. Wir verwenden in den wenigen Beispielen in diesem Heft das empfohlene `.php`.

Der PHP-Interpreter führt im nächsten Schritt zunächst eine Syntaxanalyse durch. Häufig wird in diesem Zusammenhang auch der Begriff **parsen** (engl. to parse = analysieren, aufgliedern, zergliedern) verwendet. Dabei zerlegt er den Quelltext der Datei und bringt ihn in eine für die Ausführung geeignete Form. Anschließend wird der PHP-Code ausgeführt.



Das Ergebnis liefert der Interpreter dann zurück an den Webserver, von wo es im letzten Schritt als Antwort an den Client zurückgesandt wird.

Oft wird das Ergebnis ein dynamisch erzeugtes HTML-Dokument sein. Genauso gut kann es aber auch ein PDF-Dokument, eine Grafik oder eine Datei eines anderen Formats, zum Beispiel XML oder CSV, sein. Die Möglichkeiten, die PHP hier bietet, sind vielfältig.

## 2.5 Erste Schritte mit PHP

### Vorab ein Hinweis:

Im Folgenden erarbeiten wir uns an kleinen praktischen Beispielen einen Einstieg in PHP. Diese dienen dazu, ein grundsätzliches Verständnis von der Funktionsweise serverseitiger Programmierung mit PHP zu bekommen.

Es ist im Rahmen dieses Studienhefts völlig ausreichend, wenn Sie die Beispiele theoretisch nachvollziehen.

Wenn Sie aber an serverseitiger Programmierung interessiert sind, ist es effektiver und vermutlich unterhaltsamer, wenn Sie mit PHP auch ein wenig praktisch arbeiten.

Wie Sie zuvor bereits gelernt haben, benötigen Sie dazu eine Umgebung bestehend aus mindestens einem Webserver (z. B. Apache) mit aktiviertem PHP-Modul. Sollten Sie auch Datenbankzugriffe üben wollen, muss auch noch ein RDBMS (zum Beispiel MySQL oder MariaDB) vorhanden sein.

Ohne viel Aufwand lässt sich eine solche Testumgebung einrichten. Im heft-bezogenen Downloadbereich auf der Online-Lernplattform finden Sie Lernvideos, die Ihnen die Installation solcher Umgebungen und die ersten Schritte vorstellen.

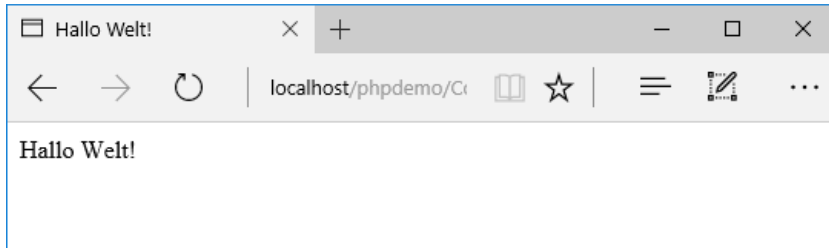
Beginnen wir also mit unserem ersten PHP-Skript.

Die Verwendung von PHP kann sehr einfach sein. Für den Anfang betten Sie PHP-Code direkt in HTML-Code ein. Sie benötigen dazu keine spezielle Entwicklungsumgebung, sondern können im einfachsten Fall einen normalen Texteditor verwenden.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Hallo Welt!</title>
</head>
<body>
  <?php
    echo "Hallo Welt!";
  ?>
</body>
</html>
```

**Code 2.1:** Hallo Welt

Zu den HTML-Tags benötigen Sie keine Erläuterungen mehr, denn Sie kennen sie bereits sehr gut. Damit PHP-Code als solcher erkannt wird, muss er zwischen `<?php` und `?>` stehen. In unserem Beispiel wird mit dem Befehl `echo` der Text "Hallo Welt!" ausgegeben. Der Befehl `echo` dient dazu, eine oder mehrere Zeichenkette(n) auszugeben.



**Abb. 2.2:** „Hallo Welt!“ im Browser

Wenn Sie den Seiten Quelltext anzeigen lassen, sieht dieser in etwa folgendermaßen aus:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Hallo Welt!</title>
</head>
<body>
Hallo Welt!</body>
</html>
```

Sie erkennen, dass alle HTML-Elemente in der PHP-Datei und im Seiten Quelltext identisch sind. Lediglich an der Stelle, an der im PHP-Bereich des PHP-Dokuments der Befehl `echo "Hallo Welt!";` stand, steht jetzt im Seiten Quelltext `Hallo Welt!`.



Schauen wir uns zur Vertiefung an diesem Beispiel noch einmal die Funktionsweise von PHP an:

Durch Aufruf der PHP-Seite im Browser (per Link oder über das Adressfeld) wird ein Request an den Webserver geschickt. Dieser sucht und findet die Datei. Wegen der Dateinamenserweiterung `.php` gibt der Webserver die Datei an den PHP-Interpreter weiter. Dieser analysiert, strukturiert und interpretiert letztlich den Dateiinhalt. In diesem Beispiel haben wir in der PHP-Datei exakt einen PHP-Bereich. Der HTML-Code außerhalb des PHP-Bereichs wird einfach in das Ausgabedokument kopiert.

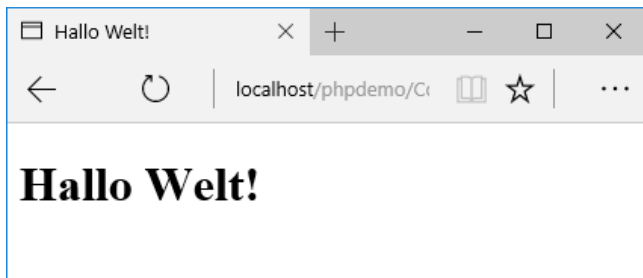
Der PHP-Code im PHP-Bereich wird ausgeführt, das heißt, die Zeichenkette "Hallo Welt!" wird an der aufrufenden Stelle ausgegeben. Anschließend gelangt das Ganze auf umgekehrtem Wege zurück zum Browser, durch den das Ergebnis als normale HTML-Datei interpretiert und angezeigt wird. Von PHP ist im Browser nichts mehr zu sehen.

Selbstverständlich können Sie auch HTML-Tags durch PHP generieren lassen. In unserem Beispiel geben wir bisher einfach einen Text in das `body`-Element aus. Das würden Sie so mit HTML bestimmt nicht machen.

Nehmen wir an, Sie möchten den Text als Überschrift erster Ordnung auszeichnen, dann passen Sie einfach Ihren `echo`-Befehl an und fügen den notwendigen HTML-Code hinzu.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Hallo Welt!</title>
</head>
<body>
  <?php
    echo "<h1>Hallo Welt!</h1>";
  ?>
</body>
</html>
```

**Code 2.2:** Hallo Welt 2



**Abb. 2.3:** Ausgabe als Überschrift

Sie mögen nun einwenden, dass Sie für die Ausgabe eines einfachen Textes kein PHP benötigen. Bedenken Sie aber: Zwischen `<?php` und `?>` kann beliebiger PHP-Code stehen und die Ausgabe des Literals `"Hallo Welt!"` war nur ein sehr, sehr einfaches Beispiel für die grundsätzliche Funktionsweise.

## 2.5.1 Rechnen

Sie können mit PHP zum Beispiel auch rechnen.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Rechnen</title>
</head>
<body>
  <?php
    $operand1 = 25;
    $operand2 = 5;

    //Addition
    echo ($operand1 + $operand2);
    //Subtraktion
    echo ($operand1 - $operand2);
```

```
//Multiplikation
echo ($operand1 * $operand2);
//Division
echo ($operand1 / $operand2);
?>
</body>
</html>
```

**Code 2.3:** Rechnen

Kurz zur Erläuterung: Wir haben hier zwei Variablen `$operand1` und `$operand2`, denen ein Wert zugewiesen wurde.

**Hinweis:**

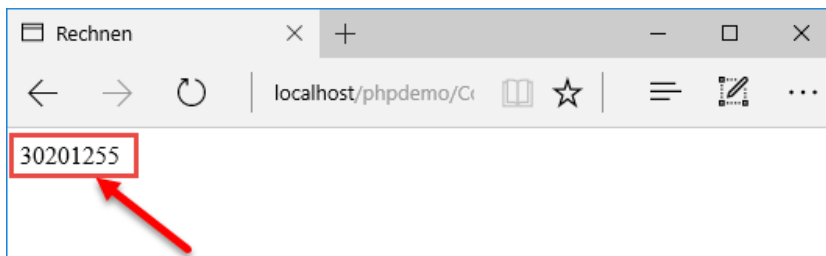
Eine Variable ist ein Datenobjekt, das seinen Wert verändern kann. Variablen werden dazu benutzt, Werte zur Programmlaufzeit zwischenspeichern.

Jede Variable hat einen Namen, den sogenannten **Bezeichner**. Im Code 2.3 haben die Variablen zum Beispiel die Bezeichner `$operand1` und `$operand2`. Über den Bezeichner einer Variablen kann auf den Wert dieser Variablen lesend und schreibend zugegriffen werden.

Der einen Variablen wurde hier der Wert 25 zugewiesen und der anderen der Wert 5. Mit diesen Werten möchten wir in den Grundrechenarten rechnen und das Ergebnis per `echo`-Befehl ausgeben. Das machen wir mit folgenden Anweisungen:

```
//Addition
echo ($operand1 + $operand2);
//Subtraktion
echo ($operand1 - $operand2);
//Multiplikation
echo ($operand1 * $operand2);
//Division
echo ($operand1 / $operand2);
```

Durch die Klammerung der Berechnung wird diese zunächst durchgeführt und dann das jeweilige Ergebnis ausgegeben. Schauen wir uns an, was dieses Skript für eine Ausgabe im Browser erzeugt.



**Abb. 2.4:** Die Ausgabe der Berechnungen im Browser

Auf den ersten Blick ist die Ausgabe nicht so, wie wir sie vielleicht erwartet oder gewünscht hätten. Anstelle der vier Ergebnisse unserer Berechnungen wird eine achtstellige Zahl angezeigt.

Beim zweiten Blick wird es aber deutlich: Da im Skript die `echo`-Anweisungen direkt hintereinanderstehen und wir auf sämtliche HTML-Tags verzichten, werden die Ergebnisse ebenfalls direkt hintereinander ausgegeben. Die achtstellige Zahl ist in dem Fall also keine achtstellige Zahl, sondern es sind die vier Ergebnisse der Berechnungen: 30, 20, 125 und 5. Diese stehen mangels Auszeichnungen lediglich direkt aneinander.

Dieses einfache Beispiel soll Ihnen verdeutlichen, dass Sie selbst dafür zuständig sind, wie die Daten ausgegeben werden. Zeilenumbrüche im PHP-Code – wie hier nach den einzelnen `echo`-Anweisungen – bleiben im erzeugten Seiten Quelltext unberücksichtigt.

Ändern wir das Beispiel noch kurz, damit die Ausgabe verständlicher wird.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Rechnen</title>
</head>
<body>
<?php
$operand1 = 25;
$operand2 = 5;

//Addition
echo "<p>Die Summe aus " . $operand1 . " und " . $operand2 . "
beträgt: " . ($operand1 + $operand2) . "</p>";
//Subtraktion
echo "<p>Die Differenz zwischen " . $operand1 . " und " . $operand2
." beträgt: " . ($operand1 - $operand2) . "</p>";
//Multiplikation
echo "<p>Das Produkt aus " . $operand1 . " und " . $operand2 . "
beträgt: " . ($operand1 * $operand2) . "</p>";
//Division
echo "<p>Der Quotient aus " . $operand1 . " durch " . $operand2 . "
beträgt: " . ($operand1 / $operand2) . "</p>";
?>
</body>
</html>
```

#### Code 2.4: Rechnen mit korrigierter Ausgabe

Im Browser sieht das Ergebnis dann zum Beispiel aus wie auf der folgenden Abbildung.

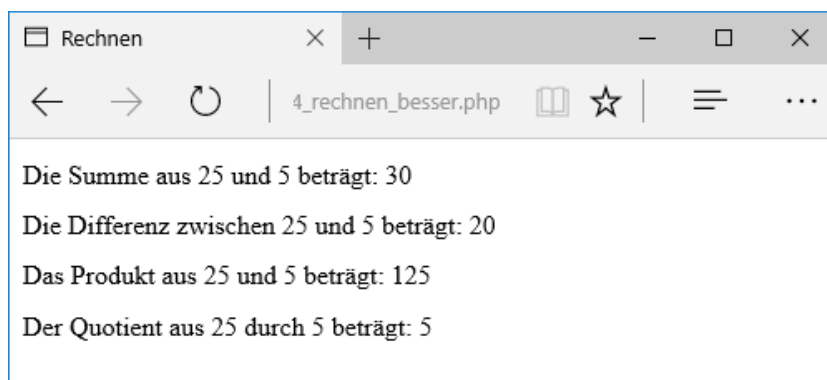


Abb. 2.5: Absatzweise Ausgaben

In jedem `echo` wird nun auch ein `<p>`-Tag ausgegeben. Das heißt, dass der Text auf der generierten Seite in einem eigenen Absatz stehen wird. Zudem enthalten die Ausgaben nun erläuternde Texte zu den Ausgaben.

#### Hinweis:

Noch zwei kurze Anmerkungen zum besseren Verständnis des PHP-Codes:

Der doppelte Schrägstrich `//` kommentiert den Rest der Zeile aus.

```
//Addition
```

Alles, was nach den Zeichen `//` in derselben Zeile steht, gilt dann als Kommentar und wird nicht ausgeführt.

Der Punkt `(.)` zwischen den einzelnen Teilen einer Zeichenkette ist der Verkettungsoperator. Damit werden mehrere Teilzeichenketten zu einer Zeichenkette verkettet.

```
echo "<p>Die Summe aus " . $operand1 . " und " . $operand2 . "
    beträgt: " . ($operand1 + $operand2) . "</p>";
```

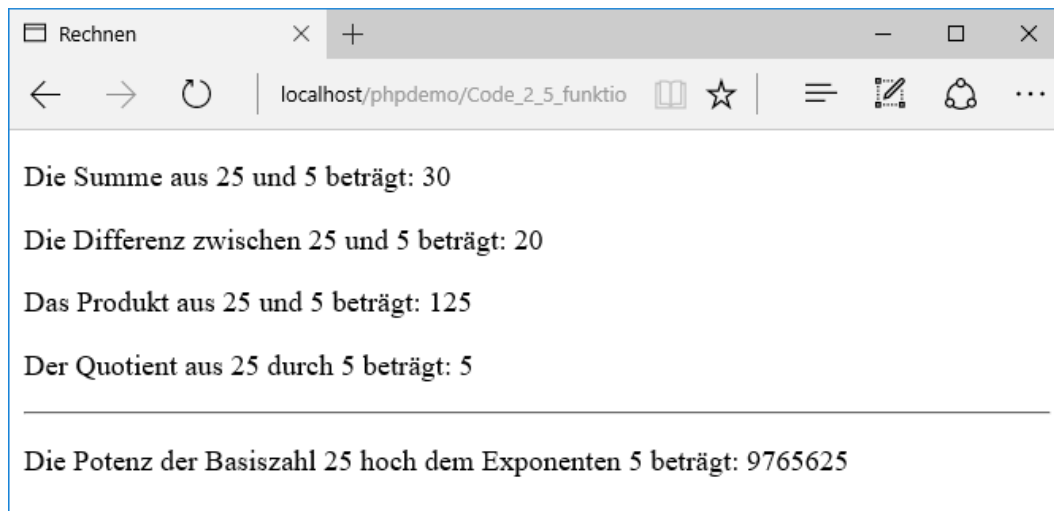
Sie können mit PHP also etwas berechnen lassen und etwas ausgeben. Dabei haben Sie den Vorteil, dass Sie das Rad nicht immer neu erfinden müssen. PHP stellt zahlreiche eingebaute (built-in) Funktionen zu verschiedenen Bereichen zur Verfügung. Diese müssen Sie nur noch benutzen. Sehen wir uns das auch in einigen Beispielen an.

Wollten wir unsere Berechnungen zum Beispiel um eine Potenzrechnung erweitern, stünde uns dafür eine PHP-Funktion zur Verfügung.

```
...
echo "<hr />";
//Exponentialrechnung
echo "<p>Die Potenz der Basiszahl " . $operand1 . " hoch dem
    Exponenten " . $operand2 . " beträgt: " . (pow($operand1,
    $operand2)) . "</p>";
...
```

#### Code 2.5: Verwendung der Funktion `pow()`

Die PHP-Funktion `pow()` liefert Ihnen das Ergebnis einer Exponentialrechnung zurück. Als Parameter erwartet die Funktion die Basiszahl und den Exponenten durch Komma getrennt in runden Klammern. In der Ausgabe sieht das für unser Beispiel dann folgendermaßen aus:



**Abb. 2.6:** Die Ausgabe der Potenz (am unteren Rand)

### 2.5.2 PHPinfo

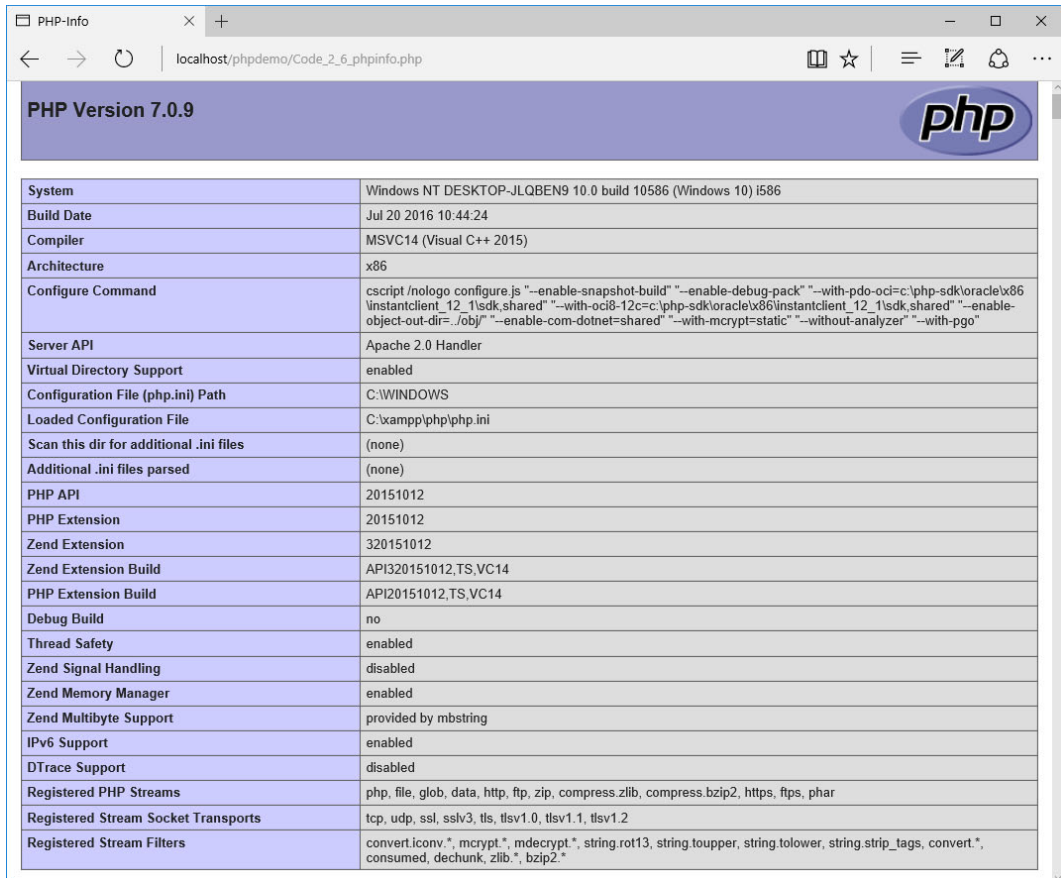
Die Anzahl der PHP-Funktionen ist fast unüberschaubar. Es gibt nicht nur mathematische Funktionen, sondern auch Funktionen aus vielen anderen Anwendungsbereichen, zum Beispiel Datums- und Uhrzeitfunktionen, Zeichenkettenfunktionen, Arrayfunktionen<sup>10</sup>, Dateifunktionen, Datenbankfunktionen und viele mehr.

Die Funktion `phpinfo()` liefert Ihnen zum Beispiel einen umfassenden Überblick über Ihre PHP-Installation.

```
...  
<?php  
  
    phpinfo();  
  
?>  
...
```

**Code 2.6:** `phpinfo()`

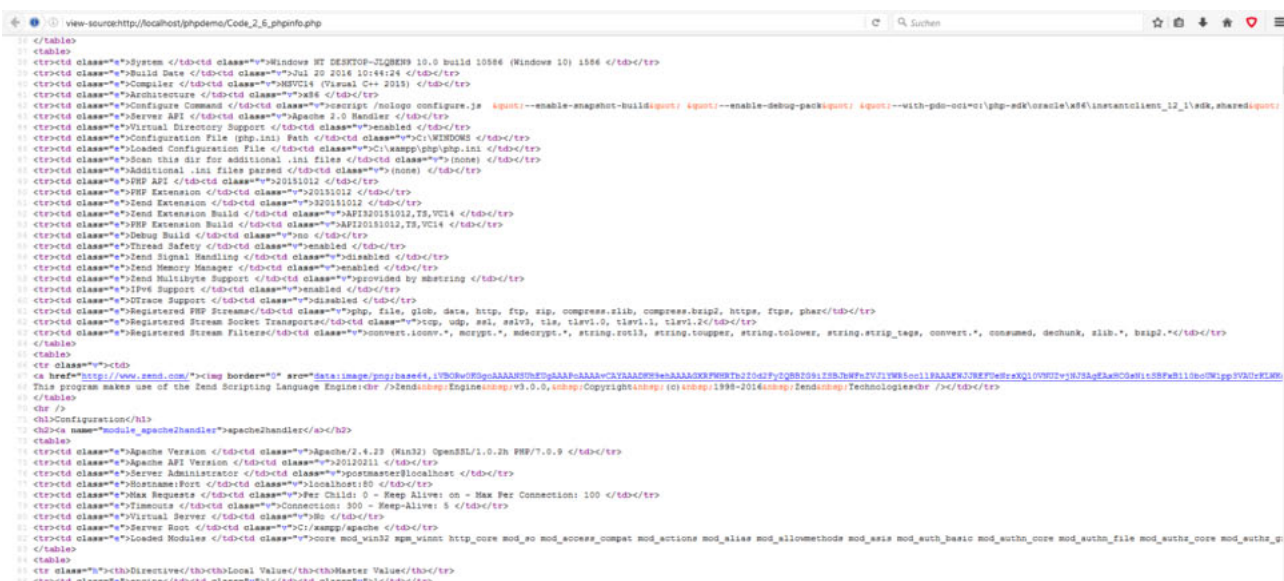
10. Ein Array ist eine Liste von Schlüssel-Wert-Paaren. Über den Schlüssel des Elements kann auf den Wert des Elements zugegriffen werden.



System	Windows NT DESKTOP-JLQBN9 10.0 build 10586 (Windows 10) i586
Build Date	Jul 20 2016 10:44:24
Compiler	MSVC14 (Visual C++ 2015)
Architecture	x86
Configure Command	cmd /c nolog configure.js --enable-snapshot-build --enable-debug-pack --with-pdo-oci=c:\php-sdk\oracle\w86\instantclient_12_1\sdk,shared --with-oci8-12c=c:\php-sdk\oracle\w86\instantclient_12_1\sdk,shared --enable-object-out-dir=.obj --enable-com-dotnet=shared --with-mcrypt=static --without-analyzer --with-pgsql
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS
Loaded Configuration File	C:\xampp\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20151012
PHP Extension	20151012
Zend Extension	320151012
Zend Extension Build	API320151012,TS,VC14
PHP Extension Build	API20151012,TS,VC14
Debug Build	no
Thread Safety	enabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	php, file, glob, data, http, ftp, zip, compress.zlib, compress.bzip2, https, ftps, phar
Registered Stream Socket Transports	tcp, udp, ssl, sslv3, tls, tlsv1.0, tlsv1.1, tlsv1.2
Registered Stream Filters	convert.iconv.*, mcrypt.*, mdecrypt.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, zlib.*, bzip2.*

Abb. 2.7: Die Ausgabe von phpinfo() im Browser

Wenn wir uns den Quelltext dieser Seite im Browser ansehen und ihn mit den wenigen Zeilen Quellcode aus Listing 2.6 vergleichen, erkennen wir schnell, dass über den Aufruf dieser einen PHP-Funktion einerseits viele zusätzliche Informationen abgerufen werden und andererseits hunderte Quelltextzeilen automatisch generiert wurden.



```

<table>
|  |  |
| --- | --- |
| System | Windows NT DESKTOP-JLQBN9 10.0 build 10586 (Windows 10) i586 |
| Build Date | Jul 20 2016 10:44:24 |
| Compiler | MSVC14 (Visual C++ 2015) |
| Architecture | x86 |
| Configure Command | cmd /c nolog configure.js --enable-snapshot-build --enable-debug-pack --with-pdo-oci=c:\php-sdk\oracle\w86\instantclient_12_1\sdk,shared --with-oci8-12c=c:\php-sdk\oracle\w86\instantclient_12_1\sdk,shared --enable-object-out-dir=.obj --enable-com-dotnet=shared --with-mcrypt=static --without-analyzer --with-pgsql |
| Server API | Apache 2.0 Handler |
| Virtual Directory Support | enabled |
| Configuration File (php.ini) Path | C:\WINDOWS |
| Loaded Configuration File | C:\xampp\php\php.ini |
| Scan this dir for additional .ini files | (none) |
| Additional .ini files parsed | (none) |
| PHP API | 20151012 |
| PHP Extension | 20151012 |
| Zend Extension | 320151012 |
| Zend Extension Build | API320151012,TS,VC14 |
| PHP Extension Build | API20151012,TS,VC14 |
| Debug Build | no |
| Thread Safety | enabled |
| Zend Signal Handling | disabled |
| Zend Memory Manager | enabled |
| Zend Multibyte Support | provided by mbstring |
| IPv6 Support | enabled |
| DTrace Support | disabled |
| Registered PHP Streams | php, file, glob, data, http, ftp, zip, compress.zlib, compress.bzip2, https, ftps, phar |
| Registered Stream Socket Transports | tcp, udp, ssl, sslv3, tls, tlsv1.0, tlsv1.1, tlsv1.2 |
| Registered Stream Filters | convert.iconv.*, mcrypt.*, mdecrypt.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, zlib.*, bzip2.* |

```

Abb. 2.8: Von phpinfo() generierter Quelltext



### 2.5.3 PHP-Code mehrfach verwenden

PHP-Code dürfen Sie an beliebigen Stellen in der PHP-Datei verwenden, und das sogar mehrfach. Sehen wir uns auch dazu ein Beispiel an. Alle PHP-Bereiche sind fett hervorgehoben.

```
<!DOCTYPE html>
<html>
<head>
<?php
//eigene Funktion, die die Zahl des Wochentages in den
//Tag umsetzt
function wochentag($wtag)
{
    switch ($wtag) {
        case 0:
            $ergebnis = "Sonntag";
            break;
        case 1:
            $ergebnis = "Montag";
            break;
        case 2:
            $ergebnis = "Dienstag";
            break;
        case 3:
            $ergebnis = "Mittwoch";
            break;
        case 4:
            $ergebnis = "Donnerstag";
            break;
        case 5:
            $ergebnis = "Freitag";
            break;
        case 6:
            $ergebnis = "Samstag";
            break;
    }
    return $ergebnis;
}

?>

<title><?php
    //Stellt im title-Tag das aktuelle Datum zur Verfügung
    //Wenn der Funktion date() kein Datum übergeben wird,
    //verwendet sie das aktuelle Serverdatum
    $datum = date("d.m.Y");
    echo $datum;
?>
</title>
<meta charset="UTF-8" />
</head>
<body>
<h1><?php
    //Stellt in der Überschrift die Uhrzeit dar
    $datum = date("H:i:s");
```

```

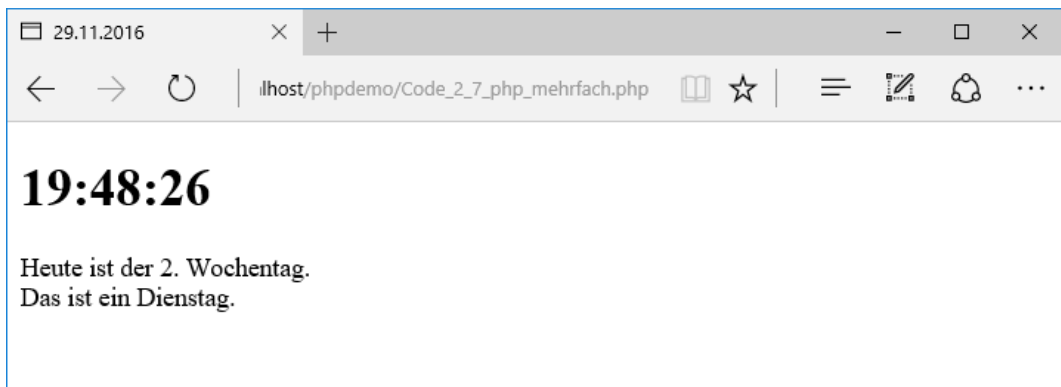
    echo $datum;
?>
</h1>

<p>Heute ist der <?php
    //ermittelt den aktuellen Tag als Zahl und gibt ihn aus
    $tag = date("w");
    echo $tag;
?>
. Wochentag.<br /> Das ist ein <?php
//ruft die eigene Funktion wochentag() auf und übergibt
//dieser die Nummer des aktuellen Tages
//liefert den Wochentag als Text zurück
    echo wochentag($tag) ;?>.</p>
</body>
</html>

```

**Code 2.7:** Verschiedener PHP-Code an mehreren Stellen

Das generierte Dokument ist auf folgender Abbildung zu sehen.

**Abb. 2.9:** Skript nach der Ausführung

Kurz zur inhaltlichen Erklärung:

Das erste Vorkommen von PHP ist die `function wochentag($wtag)`, die sich innerhalb des `Head`-Tags befindet. Sie wird zunächst nur bekannt gemacht. Als Argument wird ihr der Wochentag als Zahl übergeben. Als Rückgabewert liefert sie den Wochentag als Zeichenkette zurück. Der Aufruf dieser Funktion erfolgt erst später.

Im zweiten PHP-Abschnitt wird – etwas vereinfacht ausgedrückt – mit

```

<title>
<?php
    $datum = date("d.m.Y");
    echo $datum;
?>
</title>

```

die aktuelle Serverzeit abgerufen, als Datum formatiert und im Seitentitel ausgegeben.

Anschließend erfolgt mit

```
<h1><?php $datum = date("H:i:s");
    echo $datum;
?>
</h1>
```

die Ausgabe der Zeit im Format Stunde:Minute:Sekunde als Überschrift erster Ordnung.

Dann wird mit

```
<p>Heute ist der <?php $tag = date("w");
echo $tag;
```

einer Variablen `$tag` der Wochentag als Zahl zugewiesen und diese dann ausgegeben.

Zuletzt rufen wir mit

```
<br /> Das ist ein <?php echo wochentag($tag);?>.</p>
```

die Funktion `wochentag()` auf, die wir oben bereits bekannt gemacht haben. Als Argument übergeben wir die Variable `$tag`, deren Wert unverändert der Wochentag als Zahl ist. Den Rückgabewert der Funktion – das ist der Wochentag als Zeichenkette – geben wir dann aus.

#### Zur Erinnerung:

Dateien mit PHP-Code sollten die Namenserverweiterung **.php** tragen. So werden sie vom Parser und vom **PHP-Interpreter** korrekt verarbeitet. Außerdem müssen die Dateien in einem Verzeichnis stehen, auf das der Webserver zugreifen kann.



In einer XAMPP-Standardinstallation, wie Sie hier verwendet wird, ist das Root-Verzeichnis<sup>11</sup> des Webservers im Dateisystem zum Beispiel das Verzeichnis `C:\xampp\htdocs`. Wenn Sie also `localhost` im Adressfeld des Browsers eingeben, wird die `index.php` ausgeführt, die im Dateisystem im Verzeichnis `C:\xampp\htdocs` liegt. Für unsere Beispiele haben wir die PHP-Dateien in einem neuen Verzeichnis `phpdemo` abgelegt. Das heißt, wir rufen unsere Dokumente über <http://localhost/phpdemo/> auf.

11. Root-Verzeichnis ist ein anderer Begriff für Wurzel- oder Hauptverzeichnis und beschreibt die oberste Ebene einer Verzeichnishierarchie.

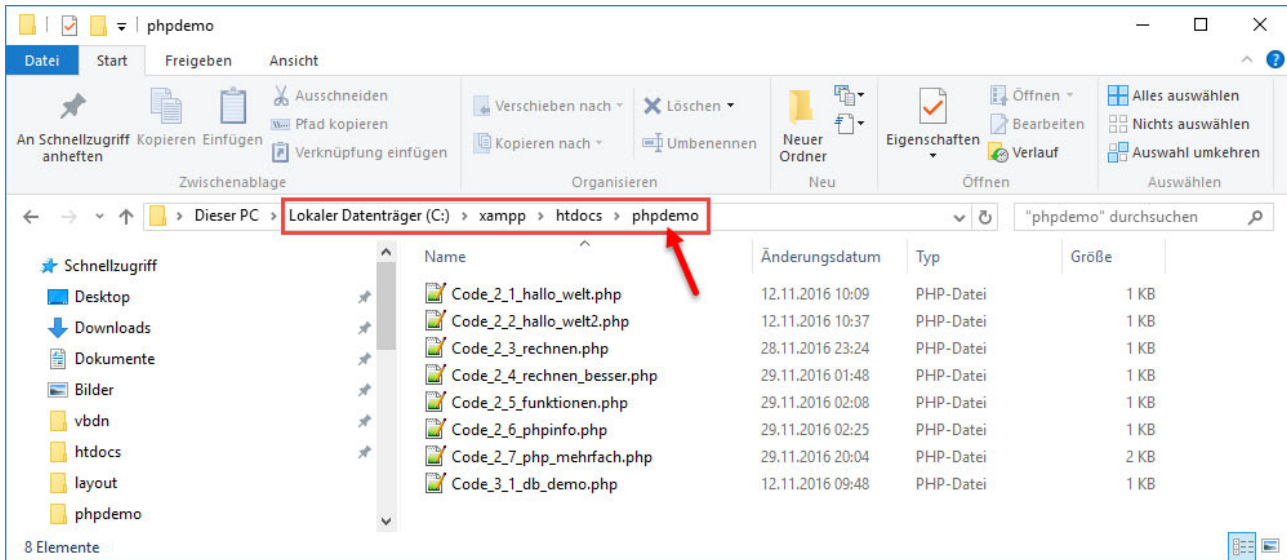


Abb. 2.10: PHP-Dateien im Dateisystem

**Bitte beachten Sie:**

Anders als reine HTML-Dateien können PHP-Dateien **nicht** über das Dateisystem ausgeführt werden, sondern benötigen immer den Aufruf über den Webserver. Nur so können Parser und Interpreter die PHP-Dateien verarbeiten.

Wenn Sie einmal ein Problem beim Ausführen eines PHP-Skripts haben, dann schauen Sie zunächst in der Adresszeile des Browsers nach. Wenn der Eintrag dort mit `file:///` gefolgt von einem Dateipfad angegeben ist, dann haben Sie die Datei nicht über den Webserver ausgeführt, sondern über das Dateisystem – beispielsweise mit einem Doppelklick im Windows Explorer.

**2.5.4 Formular auswerten**

PHP bietet mittlerweile vieles, was eine leistungsfähige Skriptsprache benötigt. Auch komplexe und sehr umfangreiche Webanwendungen wurden serverseitig mit PHP programmiert und werden das mit sehr großer Wahrscheinlichkeit auch weiterhin. Im Rahmen dieses Studienhefts, das nur in die grundlegende Funktionsweise von PHP einführen soll, ist es weder möglich noch sinnvoll, auf zu viele Details von PHP einzugehen.

Zu Beginn hatten Sie ja bereits erfahren, dass PHP erste Popularität vor allem erlangt hat, weil das Auswerten von Formularen wirklich sehr unkompliziert ist. Wie das praktisch funktionieren kann, möchten wir Ihnen noch kurz an einem einfachen Beispiel vorstellen.

Nehmen wir einmal folgendes Formular. Über zwei Auswahllisten sollen hier Speisen und Getränke ausgewählt und das Formular dann abgeschickt werden.

Essen bestellen

Bitte wählen Sie eine Speise und ein Getränk.  
Senden Sie das Formular dann bitte ab:

Frischen Salat ▾ Speisen

Mineralwasser ▾ Getränke

Bestellen

**Abb. 2.11:** Ein einfaches Formular

Der Quelltext der HTML-Datei enthält für Sie wenige Neuigkeiten.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Essen bestellen</title>
</head>
<body>
<h3>Bitte wählen Sie eine Speise und ein Getränk.<br />
Senden Sie das Formular dann bitte ab:</h3>
<form action="Code_2_9_formular_auswertung.php" method = "post">
<p>
<select name="speise">
  <option value="Burger">Burger</option>
  <option value="Schnitzel" >Schnitzel</option>
  <option value="Salat" selected="selected">Frischen Salat </
  option>
</select> Speisen
</p>
<p>
<select name="getraenk">
  <option value="Coke">Coke</option>
  <option value="Wasser" selected="selected">Mineralwasser</
  option>
  <option value="Tee">Tee</option>
</select> Getränke
</p>
<p>
  <input type="submit" value="Bestellen"/>
</form>
</body>
</html>
```

**Code 2.8:** Das Formular

Um das Formular mit PHP auszuwerten, müssen Sie dem Attribut `Action` den Namen der PHP-Datei zuweisen, die die Auswertung durchführen soll.

In diesem Beispiel ist das die Datei `Code_2_9_formular_auswertung.php`.

**Hinweis:**

Die Auswertung erfolgt nicht zwangsläufig – wie in diesem Beispiel – in einer anderen Datei. Ganz im Gegenteil: Häufig wird die Auswertung in der Datei durchgeführt, in der auch das Formular steht.

Die Methode zum Übertragen des Formular ist hier `post`, erkennbar am Wert des Attributs `method`.

Mit `get` könnten die Daten aber auch gesendet und anschließend durch PHP verarbeitet werden.

HTML-Formulare und die Methoden zum Übertragen sind Ihnen bereits geläufig. Wir müssen hier auf die einzelnen Unterschiede zwischen **post** und **get** nicht noch einmal im Detail eingehen.

**Zur Erinnerung:**

Bei der Methode **get** werden die Formulardaten mit dem URL übertragen.  
Bei **post** dagegen in der Nachricht selbst.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>Essen bestellen - Auswertung</title>
</head>
<body>
<?php
$essen = $_POST["speise"];
$trinken = $_POST["getraenk"];

echo "Sie möchten also " . $essen . " essen<br />";
echo " und dazu " . $trinken . " trinken!<br />";

$essenpreis = 0;
$trinkenpreis = 0;

switch ($essen) {
  case "Burger":
    $essenpreis = 8.95;
    break;
  case "Schnitzel":
    $essenpreis = 10.95;
    break;
  case "Salat":
    $essenpreis = 7.50;
    break;
}

switch ($trinken) {
  case "Coke":
    $trinkenpreis = 3.20;
    break;
```

```

    case "Wasser":
        $trinkenpreis = 2.50;
        break;
    case "Tee":
        $trinkenpreis = 3.20;
        break;
}

echo "<p>Das macht € " . ($essenpreis + $trinkenpreis) . "</p>";
?>
</body>
</html>

```

**Code 2.9:** Die Auswertung des Formulars

Schauen wir uns die Auswertung der Formulardaten durch PHP jetzt einmal an.

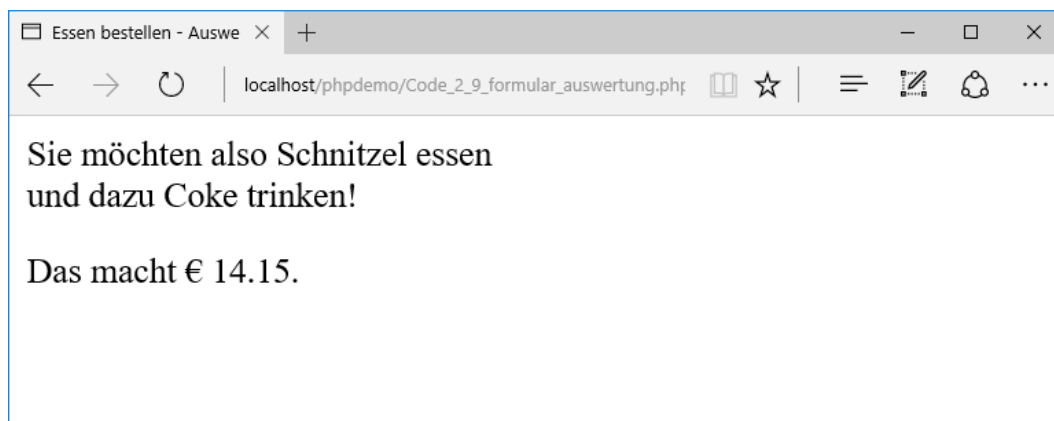
Die gesendeten Formulardaten stellt PHP über ein vordefiniertes superglobales<sup>12</sup> assoziatives Array<sup>13</sup> zur Verfügung. Werden die Daten wie bei uns mit der Methode `post` gesandt, dann verwenden Sie zur Auswertung `$_POST`, bei `get` entsprechend `$_GET`. Um auf den Inhalt eines Formularfeldes zuzugreifen, geben Sie dahinter in eckigen Klammern dessen Namen in Anführungszeichen an.

```

$essen = $_POST["speise"];
$trinken = $_POST["getraenk"];

```

Über diese beiden Zeilen lesen wir in unserem Beispiel also aus, was in den Formularfeldern mit den Namen `speise` und `getraenk` ausgewählt wurde, und weisen diese Werte an lokale Variablen zu.

**Abb. 2.12:** Die Auswertung des Formulars

Im Browser sieht die Ausgabe der Auswertung aus wie in der Abbildung oben.

So viel zur allgemeinen Funktionsweise von PHP und zur Formularauswertung mithilfe der assoziativen Arrays `$_GET` und `$_POST`.

12. Superglobals sind in PHP Variablen, die in allen Gültigkeitsbereichen zur Verfügung stehen. Sie können von jeder Stelle in dem PHP-Skript darauf zugreifen.

13. Unter einem assoziativen Array versteht man ein Array, auf dessen Elemente über Namen und nicht über einen numerischen Index zugegriffen wird.

## Zusammenfassung

PHP steht für PHP Hypertext Preprocessor. PHP ist relativ leicht zu erlernen, ausgereift und extra für die Entwicklung von Webanwendungen optimiert.

PHP wird auf dem Server ausgeführt.

PHP-Code kann direkt in HTML-Code eingebettet werden. PHP-Code wird vom Interpreter zunächst geparkt und dann verarbeitet.

PHP unterstützt seit der Version 5 vollständig objektorientierte Programmierung.

PHP-Code kann beliebig oft und an beliebigen Stellen in den HTML-Code eingebettet werden.

Auf Formulardaten greifen Sie mit PHP über die Elemente der superglobalen assoziativen Arrays `$_POST` oder `$_GET` zu.

---

## Aufgaben zur Selbstüberprüfung

- 2.1 Notieren Sie die Zeichen, mit denen Sie einen PHP-Block öffnen, und die, mit denen Sie ihn wieder schließen.
  - 2.2 Welche Aufgabe erfüllt der PHP-Interpreter?
  - 2.3 Warum müssen Dateien mit PHP-Code die Dateinamenerweiterung `.php` haben?
  - 2.4 Wie lautet ein PHP-Kommando, um etwas auszugeben?
-



### 3 MySQL

*In dieser Lektion erfahren Sie etwas über MySQL und Sie lernen, wie PHP und MySQL zusammenarbeiten.*

Wenn es – wie in diesem Studienheft – um die Grundlagen von Webanwendungen und serverseitige Programmierung geht, dann müssen wir auch einen kurzen Blick auf Datenbanksysteme werfen.

Kaum eine Webanwendung kommt heutzutage noch ohne die Anbindung an eine Datenbank aus. Wenn einer Anwendung größere Datenmengen zur Verfügung gestellt werden müssen, Daten durch Benutzer bearbeitet oder hinzugefügt werden können, sind in der Regel **Datenbankmanagementsysteme** (DBMS) im Einsatz.

Die Anforderungen an solche Systeme sind hoch. Sie müssen schnell, sicher, flexibel und vor allen Dingen stabil sein.

PHP bietet Schnittstellen zu über zwanzig verschiedenen solcher Systeme. In der Praxis hat sich besonders die Kombination aus PHP und MySQL etabliert.

**MySQL** ist ein **relationales Datenbankmanagementsystem**<sup>14</sup> (RDBMS), das seit Mitte der 1990er-Jahre vom Unternehmen MySQL AB entwickelt wurde. Nach zweimaliger Übernahme gehört MySQL seit 2010 zu Oracle. Neben einigen kommerziellen Lizenzen gibt es MySQL in der **Community Edition** als Open-Source-Software. MySQL ist sehr weit verbreitet und in vielen Hosting-Angeboten inkludiert.

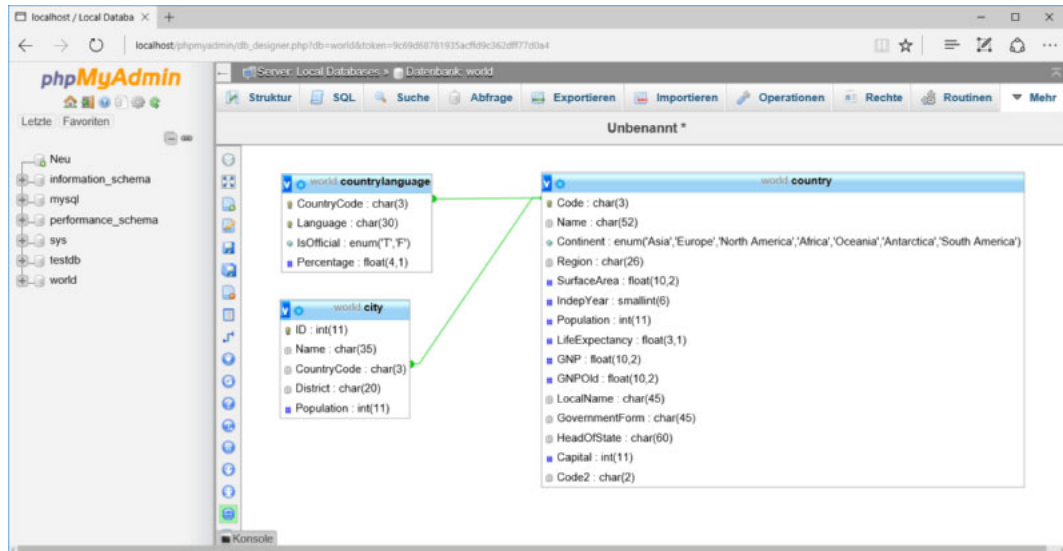
#### Hinweis:

Bereits 2009 erschien mit **MariaDB** eine Abspaltung – ein sogenannter Fork – des MySQL-Projekts, der aufgrund des Lizenzierungsmodells, der aktiven Community und der stetigen Weiterentwicklung zunehmend an Akzeptanz und Bedeutung gewinnt.

MariaDB ist in vielen Teilen kompatibel mit MySQL und verwendet zum Beispiel auch dieselben Schnittstellen zu PHP.

In RDBMS wie MySQL werden die Daten in Tabellen organisiert, die in einer bestimmten Beziehung zu anderen Tabellen derselben Datenbank stehen können.

14. RDBMS basieren auf dem relationalen Datenmodell. Dabei werden die Daten in Tabellenform – den sogenannten Relationen – abgelegt. Die Spalten der Tabelle – im relationalen Datenmodell Attribute genannt – beschreiben dabei die Eigenschaften der Daten, also zum Beispiel den Namen, den Vornamen oder die Straße.



**Abb. 3.1:** Struktur und Beziehungen von Tabellen einer Datenbank in phpMyAdmin<sup>15</sup>

Ein RDBMS kann mehrere Datenbanken verwalten. Jede dieser Datenbanken kann wiederum eine Vielzahl von Tabellen enthalten.

Wenn möglich, sollten Sie pro Projekt eine eigene Datenbank anlegen. Das ist praktisch und übersichtlich.

#### Hinweis:

Oft wird die Anzahl der Datenbanken durch Hosting-Anbieter stark limitiert. Wenn Sie beispielsweise nur eine Datenbank zur Verfügung haben, sollten Sie die Tabellen eines Projekts mit einem einheitlichen Präfix<sup>16</sup> versehen. Alle Tabellen, die innerhalb derselben Datenbank diese Kennung tragen, lassen sich dann leicht als zusammengehörig identifizieren.

MySQL verwendet – wie viele andere RDBMS auch – die Datenbanksprache **SQL** (Structured Query Language)<sup>17</sup>. SQL stellt Befehle zur Datenbankverwaltung, zum Anlegen einer Datenbank sowie zum Erstellen, Ändern und Löschen von Tabellen und Daten zur Verfügung. Außerdem können Sie **Abfragen** auf den Datenbestand durchführen, um nach bestimmten Kriterien gefilterte Daten zu erhalten.

MySQL unterstützt den SQL-Standard nicht vollständig, bietet aber zusätzliche proprietäre<sup>18</sup> SQL-Erweiterungen an.

15. phpMyAdmin ist ein sehr weit verbreitetes browserbasiertes Werkzeug zum Arbeiten mit MySQL-Datenbanken.

16. Ein Präfix ist in diesem Kontext eine einheitliche Erweiterung, die dem eigentlichen Tabellennamen vorangestellt wird.

17. Dt. strukturierte Abfragesprache, manchmal wird für das „S“ auch „Standard“ anstatt „Structured“ verwendet.

18. Proprietär bedeutet im Kontext von Software so viel wie herstellerspezifisch.

### 3.1 Vorteile von MySQL

Sehen wir uns einige positive Eigenschaften von MySQL im Überblick an:

- MySQL steht für viele Betriebssysteme zur Verfügung – zum Beispiel Linux, Windows, OSX sowie Solaris.
- MySQL unterstützt zahlreiche Programmiersprachen. Eine Auswahl: C, C#, C++, Java, Perl, Python und selbstverständlich PHP.
- Verschlüsselte Passwörter und ein flexibles, mehrstufiges Berechtigungskonzept sorgen u. a. dafür, dass MySQL als sicher gilt.
- Durch die Unterstützung von SQL ist MySQL mit einfachen Mitteln zu bedienen. Einige simple SQL-Statements reichen bereits aus, um Daten aus MySQL-Datenbanken abzufragen oder zu bearbeiten.
- MySQL ist in der Community-Edition frei verfügbar.
- MySQL ist Bestandteil vieler Hosting-Angebote.
- MySQL gilt als schnell.

### 3.2 Zusammenarbeit zwischen PHP und MySQL

Sie haben zuvor erfahren, dass Sie MySQL-Datenbanken mit SQL abfragen können und dass Sie für die Verbindung zum MySQL-Server und den Zugriff auf die Daten über ausreichende Berechtigungen verfügen müssen. Aber wie funktioniert das praktisch? In unserem Fall selbstverständlich mit PHP.

Für die Zusammenarbeit mit MySQL stellte PHP bisher zwei Erweiterungen zur Verfügung: MySQL und MySQLi<sup>19</sup>. **MySQLi** ist die etwas modernere Erweiterung, die sowohl prozedurale Programmierung als auch objektorientierte Programmierung unterstützt.

Mit PHP 7 wurde die Unterstützung von MySQL eingestellt. Zudem gibt es mit **PDO** (PHP Data Objects) noch eine weitere PHP-Erweiterung, mit der Sie auf verschiedene Datenbanksysteme zugreifen können. Neben MySQL zum Beispiel auch auf MS SQL Server, Oracle, IBM Informix oder IBM DB2.

PDO bietet Abstraktion für den **Datenzugriff**, sodass Sie unabhängig von der verwendeten Datenbank dieselben Funktionen verwenden können, um Abfragen zu erstellen. PDO bietet aber **keine** Abstraktion für **Datenbanken**. Es werden also keine SQL-Anweisungen automatisch an ein bestimmtes Datenbanksystem angepasst, um beispielsweise Inkompatibilitäten in der Unterstützung von SQL zu reparieren oder Ähnliches.



19. Das i in MySQLi steht für engl. improved (dt.: verbessert)

Die Schnittstelle zwischen PHP und dem MySQL-Server besteht bei Verwendung der PHP-Erweiterung *MySQLi* aus drei Klassen:

- eine für den **Verbindungsaufbau**,
- eine weitere für die **Ergebnisse von Abfragen** und/oder
- eine dritte für **vorbereitete Anweisungen**. Sie erinnern sich, dass wir im Abschnitt „Sicherheit“ vorbereitete Anweisungen als ein Mittel gegen SQL-Injections genannt hatten.

#### Hinweis:

Der Begriff der **Klasse** stammt übrigens aus der objektorientierten Programmierung. Wir wollen an dieser Stelle nicht weiter darauf eingehen. Vielleicht so viel: Sie können sich eine Klasse als einen Bauplan vorstellen, nach dessen Vorgaben Objekte erzeugt werden können.

### 3.3 Ablauf

Der Ablauf einer Zusammenarbeit von PHP und MySQL ist immer ähnlich und besteht aus folgenden Schritten

1. Verbindung zum MySQL-Server herstellen, Datenbank auswählen und Anmeldeinformationen senden
2. Abfrage(n) senden
3. Ergebnis(se) verarbeiten, aufbereiten und ausgeben
4. Verbindung schließen

Das folgendes einfache PHP-Skript, das Sie nicht vollständig bis ins Detail verstehen müssen, verdeutlicht, wie einfach die Zusammenarbeit funktioniert.

In diesem Beispiel gibt es eine Datenbank mit dem Namen `world` auf dem Datenbankhost `localhost`. Diese Datenbank enthält eine Tabelle `country`<sup>20</sup>. In der Tabelle sind verschiedene Informationen zu 239 Ländern in englischer Sprache gespeichert.

Wir ermitteln die Anzahl der Länder und geben diese aus. Anschließend geben wir für jedes der 239 Länder den Namen, den Kontinent, zu dem es gehört, und die ungefähre Einwohnerzahl aus.

Das hört sich nach viel Arbeit an, ist es aber nicht.

20. Die Tabelle `country` stammt aus der Datenbank `world`, die häufig für Trainings und Zertifizierungen im Bereich MySQL eingesetzt wird. Diese und wenige andere Beispieldatenbanken können unter <http://dev.mysql.com/doc/index-other.html> heruntergeladen werden. Bitte beachten Sie aber die jeweiligen Lizenzbedingungen.

```

<?php
// 1. Verbindung aufbauen
$mysqli = new mysqli("localhost", "root", "", "world");

if ($mysqli->connect_error) {
    echo "Verbindungsfehler: " . mysqli_connect_error();
    exit();
}

if (!$mysqli->set_charset("utf8")) {
    exit("Zeichensatz UTF-8 konnte nicht gesetzt werden!");
}

// 2. Abfrage(n) senden
$sql = 'SELECT * FROM country';
$ergebnis = $mysqli->query($sql);
// 3. Ergebniss(e) verarbeiten und ausgeben
echo "<p>Es wurden " . $ergebnis->num_rows
    . " Länder gefunden.</p>";

while ($zeile = $ergebnis->fetch_object()) {
    echo "Das Land " . $zeile->Name
        . " gehört zu " . $zeile->Continent
        . ". Es hat ca. " . $zeile->Population. " Einwohner.<br />";
}

$ergebnis->close();
// 4. Verbindung schließen
$mysqli->close();
?>

```

**Code 3.1:** Beispiel für die vier Schritte bei der Zusammenarbeit zwischen PHP und MySQL

Auch hier eine kurze Erläuterung des Codes zu Ihrer Information. Sie müssen sich das nicht im Detail merken. Nur damit Sie grundsätzlich wissen, wie der Ablauf ist.

Der Verbindungsaufbau erfolgt mit der Anweisung

```
$mysqli = new mysqli("localhost", "root", "", "world");
```

Konkret wird hier ein Objekt der Klasse `mysqli` instanziiert. Dem Konstruktor<sup>21</sup> der Klasse werden vier Argumente übergeben:

1. der Name des Datenbankhosts – hier `localhost`
2. der Benutzername des Datenbankbenutzers – hier `root`
3. das Kennwort dieses Benutzers – hier ist keines vergeben, deswegen `""`, und
4. der Name der Datenbank – hier die Datenbank `world`.

21. Der Konstruktor ist die Methode einer Klasse, die für das Erstellen eines Objekts dieser Klasse zuständig ist.

**Hinweis:**

In der Praxis werden Sie weder mit dem Benutzer `root` noch ohne Passwort und schon gar nicht in der Kombination aus beidem Datenbankverbindungen aufbauen. Das wäre viel zu riskant und sicherheitskritisch. Das ist lediglich hier in einer lokalen Testumgebung erlaubt.

Das erzeugte Objekt wird in der Objektvariablen `$mysqli` gespeichert.

Danach wird in unserem PHP-Skript die Verbindung geprüft und der Zeichensatz der Verbindung gesetzt. Beides ist im Ablauf zwar von Bedeutung, wir gehen hier aber nicht näher darauf ein.

Nach erfolgreichem Verbindungsaufbau wird mit den Anweisungen

```
$sql = 'SELECT * FROM country';
$ergebnis = $mysqli->query($sql);
```

zunächst eine Variable `$sql` mit der SQL-Abfrage erstellt. Diese wird der Methode `query()` unseres zuvor bereits erstellten Objekts `$mysqli` als Argument übergeben.

Wenn die Abfrage erfolgreich war, dann wird als Rückgabewert von `query()` ein Objekt erzeugt<sup>22</sup>, das in der Objektvariablen `$ergebnis` gespeichert wird. Die Variable `$ergebnis` enthält jetzt die Ergebnismenge. Das bedeutet: alles, was die SQL-Abfrage zurückliefert.

Das tatsächliche Ergebnis ist dabei abhängig von der SQL-Abfrage. Das kann zum Beispiel ein einziges Feld eines Datensatzes sein, ein kompletter Datensatz, sehr viele komplette Datensätze oder auch nur eine Bestätigung, dass die Abfrage erfolgreich ausgeführt wurde. Die Möglichkeiten sind vielfältig.

In unserem Beispiel haben wir alle Felder aller Datensätze der Tabelle `country` abgefragt und erwarten eine entsprechende Ergebnismenge aus mehreren Datensätzen mit mehreren Feldern.

In einer Schleife werden diese Datensätze aus der Ergebnismenge abgeholt und ausgegeben. Dafür gibt es verschiedene Ansätze.

Zuletzt müssen sowohl das Objekt mit der Ergebnismenge als auch das Verbindungsobjekt wieder freigegeben werden. Das passiert mit den Anweisungen

```
//Ergebnismenge freigeben
$ergebnis->close();
// 4. Verbindung schließen
$mysqli->close();
```

Im Browser Edge sieht das Ergebnis dieses Skripts wie auf der folgenden Abbildung aus.

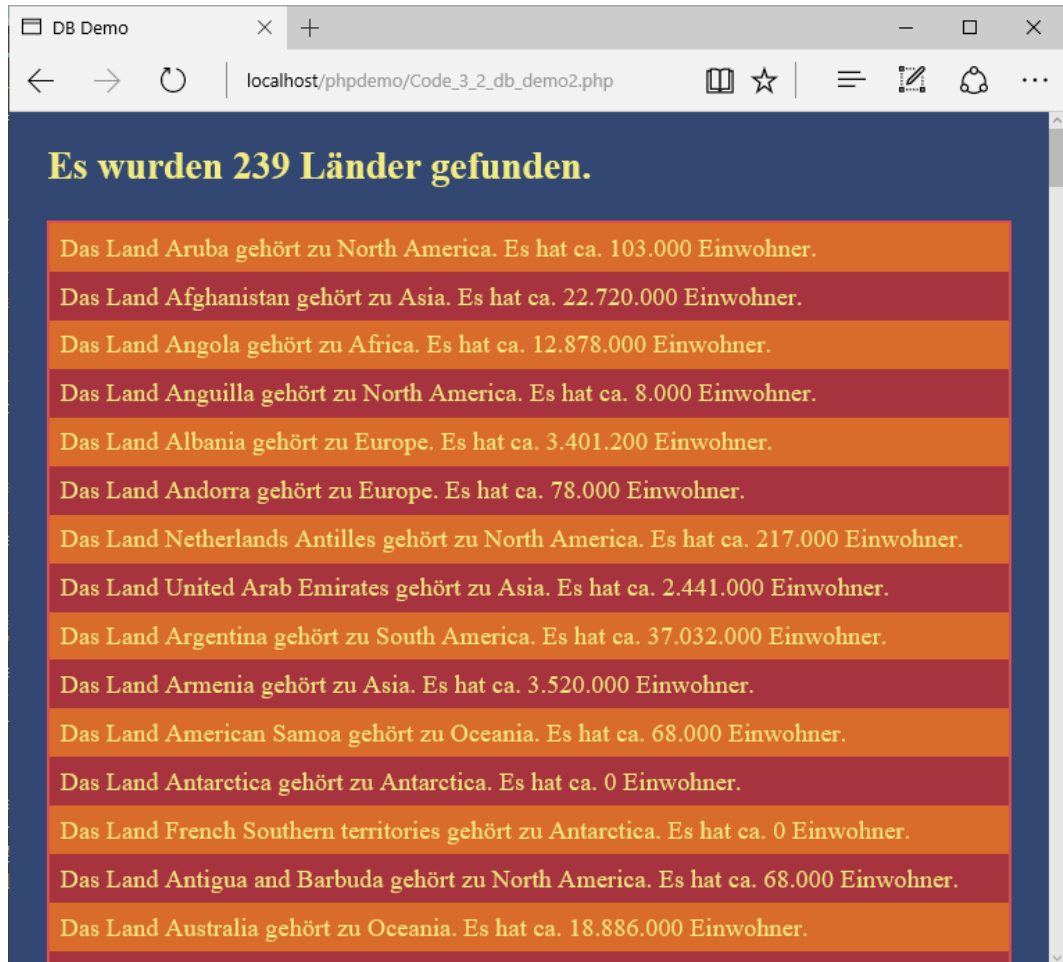
22. Dieses Objekt ist vom Typ `mysqli_result`.



**Abb. 3.2:** Das Ergebnis der Zusammenarbeit: Daten von über 200 Ländern

Falls Ihnen die sehr einfache Darstellung nicht gefällt, können Sie die Ausgaben selbstverständlich genauso mit HTML auszeichnen und mit CSS formatieren, wie Sie das von HTML-Seiten bereits kennen.





**Abb. 3.3:** Formatierte Ausgabe des Ergebnisses von zuvor

Fassen wir noch einmal zusammen:

## Zusammenfassung

MySQL ist ein relationales Datenbankmanagementsystem (RDBMS). In RDBMS werden die Daten in Tabellen gespeichert, die in einer bestimmten Beziehung zu anderen Tabellen derselben Datenbank stehen können.

MySQL unterstützt SQL, eine Datenbankabfragesprache, die u. a. Befehle zum Anlegen, Bearbeiten und Löschen von Datenbanken, Tabellen, Beziehungen und Daten zur Verfügung stellt.

Für die Zusammenarbeit mit MySQL stellt PHP die Erweiterungen MySQL, MySQLi und PDO zur Verfügung. MySQLi ist neuer als MySQL und erlaubt auch objektorientierte Programmierung. PDO bietet eine Abstraktionsschicht für den Datenzugriff auf verschiedene DBMS.

Der Ablauf einer Zusammenarbeit von PHP und MySQL besteht im Wesentlichen aus vier Schritten: Verbindung herstellen, Abfrage senden, Ergebnis verarbeiten und Verbindung schließen.



### Aufgaben zur Selbstüberprüfung

- 3.1 Was ist MySQLi und wofür steht das i?
- 3.2 Wofür steht die Abkürzung RDBMS?
- 3.3 Was unterscheidet die Erweiterung PDO konzeptionell von den Erweiterungen MySQL und MySQLi?

## 4 Editoren, IDEs und Frameworks

*In dieser Lektion lernen Sie Entwicklungswerkzeuge für PHP-Anwendungen kennen. Sie erfahren, welche Vorteile die Verwendung integrierter Entwicklungsumgebungen hat.*

*Außerdem erhalten Sie grundlegende Informationen zu PHP-Frameworks.*

Wir hatten zuvor bereits festgestellt, dass PHP-Code einfach in HTML-Quellcode eingebettet werden kann. Das erfolgt im einfachsten Fall mit einem ganz normalen Texteditor wie Notepad, dem Texteditor von Windows. Das funktioniert, ist aber wenig komfortabel.

Spätestens, wenn Sie sich vornehmen, tiefer in das Thema Webanwendungen einzusteigen, regelmäßig zu entwickeln oder Sie gar ins Auge fassen, dies professionell zu tun, dann sollten Sie sich über geeignetere Entwicklungswerkzeuge Gedanken machen.

Mit zunehmender Entwicklungserfahrung werden Ihre Ansprüche an die Qualität Ihrer Webprojekte steigen. Es wird für Sie vermutlich nicht mehr ausreichend sein, dass die Anwendung „läuft“. Vielmehr sollte sie möglichst fehlerfrei, leicht wartbar, schnell und ordentlich dokumentiert sein. Und im besten Fall soll die Entwicklungsarbeit sogar Spaß machen!

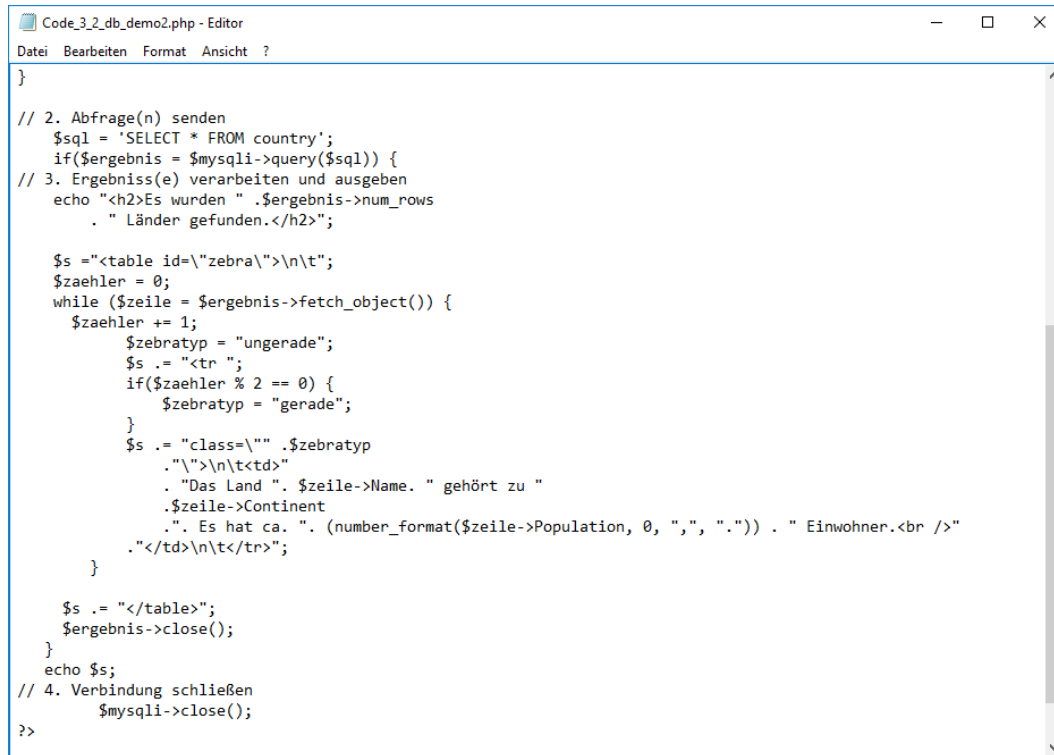
Wenn Sie sich bei der Arbeit allerdings ständig über die Unzulänglichkeiten Ihrer Werkzeuge ärgern müssen, macht die Entwicklung sicher keinen Spaß.

Gute Editoren und vor allen Dingen auch gute integrierte Entwicklungsumgebungen unterstützen Sie sehr beim Erreichen der zuvor genannten Ziele. Schauen wir uns im Folgenden ein paar davon an. Beginnen wir mit den Editoren.

### 4.1 Editoren

Editoren dienen dazu, Dateien anzulegen, zu bearbeiten und zu speichern. Im Zusammenhang mit der Codierung in PHP können verschiedene Editoren unterschieden werden.

Einfache Texteditoren – wie der zuvor genannte Editor von Windows – ermöglichen zwar die Eingabe von Text, bieten aber darüber hinaus keine spezielle Unterstützung für die Codierung. Das heißt, Sie werden weder bei der Eingabe des Codes unterstützt noch auf Fehler bei der Eingabe hingewiesen; sprachspezifische Begriffe wie Schlüsselwörter, Bezeichner oder Kommentare werden ebenfalls nicht hervorgehoben.



```

Code_3_2_db_demo2.php - Editor
Datei Bearbeiten Format Ansicht ?

}

// 2. Abfrage(n) senden
$sql = 'SELECT * FROM country';
if($ergebnis = $mysqli->query($sql)) {
// 3. Ergebniss(e) verarbeiten und ausgeben
echo "<h2>Es wurden " . $ergebnis->num_rows
    . " Länder gefunden.</h2>";

    $s = "<table id=\"zebra\">\n\t";
    $zaehler = 0;
    while ($zeile = $ergebnis->fetch_object()) {
        $zaehler += 1;
        $zebratyp = "ungerade";
        $s .= "<tr ";
        if($zaehler % 2 == 0) {
            $zebratyp = "gerade";
        }
        $s .= "class=\"\" " . $zebratyp
            . ">\n\t<td>"
            . "Das Land ". $zeile->Name. " gehört zu "
            . $zeile->Continent
            . ". Es hat ca. " . (number_format($zeile->Population, 0, ",", ".")) . " Einwohner.<br />"
            . "</td>\n\t</tr>";
    }

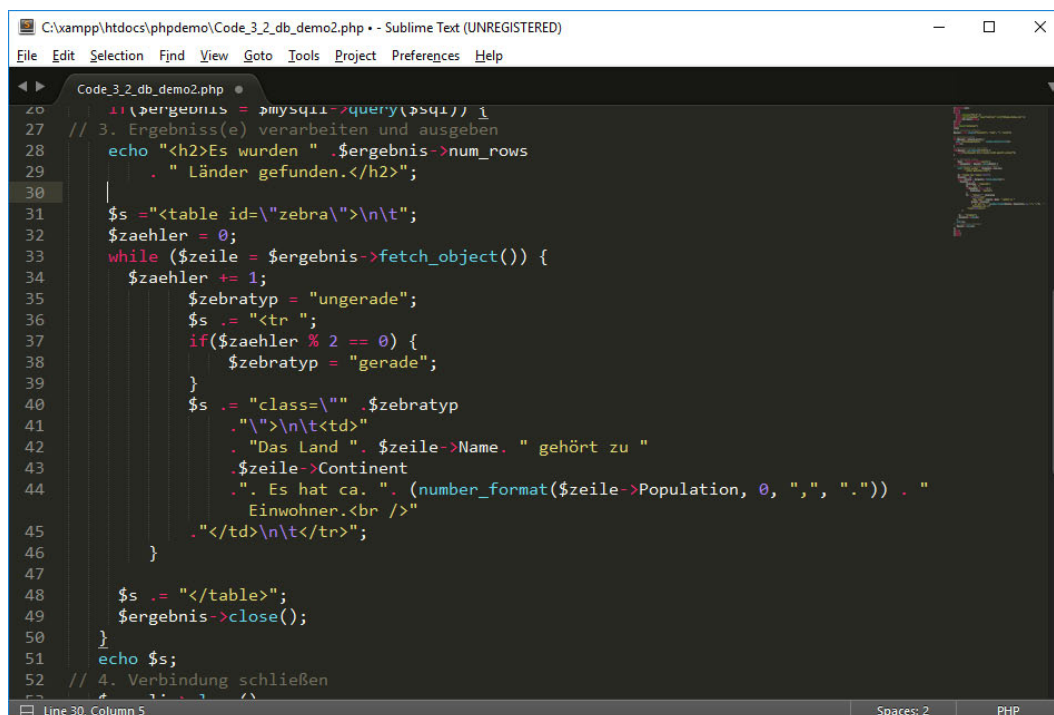
    $s .= "</table>";
    $ergebnis->close();
}
echo $s;
// 4. Verbindung schließen
$mysqli->close();
?>

```

Abb. 4.1: PHP-Datei im Windows Editor

Besser geeignet sind Editoren, die Funktionen zur Quellcodebearbeitung zur Verfügung stellen, um Ihnen die Eingabe zu erleichtern und Sie beim Suchen von Eingabefehlern zu unterstützen.

Leistungsfähigere Editoren sind beispielsweise Notepad++ oder Sublime Text.



```

C:\xampp\htdocs\phpdemo\Code_3_2_db_demo2.php - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

Code_3_2_db_demo2.php
26 // 2. Abfrage(n) senden
27 $sql = 'SELECT * FROM country';
28 if($ergebnis = $mysqli->query($sql)) {
29 // 3. Ergebniss(e) verarbeiten und ausgeben
30 echo "<h2>Es wurden " . $ergebnis->num_rows
31     . " Länder gefunden.</h2>";
32
33 $s = "<table id=\"zebra\">\n\t";
34 $zaehler = 0;
35 while ($zeile = $ergebnis->fetch_object()) {
36     $zaehler += 1;
37     $zebratyp = "ungerade";
38     $s .= "<tr ";
39     if($zaehler % 2 == 0) {
40         $zebratyp = "gerade";
41     }
42     $s .= "class=\"\" " . $zebratyp
43         . ">\n\t<td>"
44         . "Das Land ". $zeile->Name. " gehört zu "
45         . $zeile->Continent
46         . ". Es hat ca. " . (number_format($zeile->Population, 0, ",", ".")) . "
47         Einwohner.<br />"
48         . "</td>\n\t</tr>";
49     }
50
51     $s .= "</table>";
52     $ergebnis->close();
53 }
54 echo $s;
55 // 4. Verbindung schließen
56 $mysqli->close();

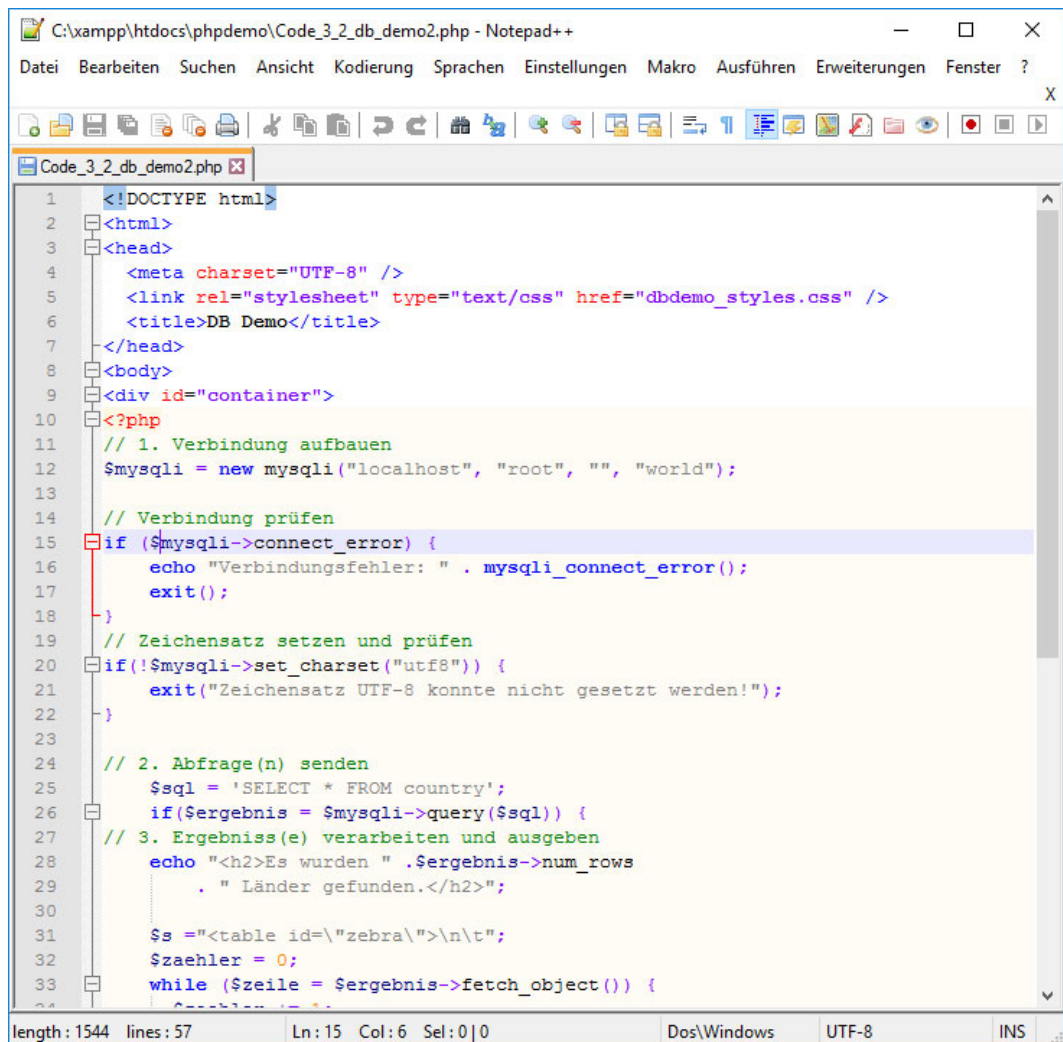
```

Abb. 4.2: Geöffnete PHP-Datei in Sublime Text

Die Features unterscheiden sich im Detail von Editor zu Editor. In der Regel lassen sich diese über Plugins auch noch erheblich im Funktionsumfang erweitern.

Typische Funktionen eines leistungsfähigen Editors sind zum Beispiel die **Syntax-hervorhebung** (Syntax-Highlighting), das **Autovervollständigen** von Wörtern, Befehlen und Funktionen (Auto-Completion) oder das Vorschlagen von Befehlen und das Anzeigen von Funktionsparametern während der Eingabe (Code-Hinting).

Wie zum Beispiel Syntax-Highlighting dabei konkret erfolgt, ist einerseits vom Editor abhängig (Was kann der Editor überhaupt?) und lässt sich andererseits auch individuell konfigurieren (Wie lässt es sich darstellen?).



```

1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8" />
5      <link rel="stylesheet" type="text/css" href="dbdemo_styles.css" />
6      <title>DB Demo</title>
7  </head>
8  <body>
9      <div id="container">
10     <?php
11         // 1. Verbindung aufbauen
12         $mysqli = new mysqli("localhost", "root", "", "world");
13
14         // Verbindung prüfen
15         if ($mysqli->connect_error) {
16             echo "Verbindungsfehler: " . mysqli_connect_error();
17             exit();
18         }
19         // Zeichensatz setzen und prüfen
20         if (!$mysqli->set_charset("utf8")) {
21             exit("Zeichensatz UTF-8 konnte nicht gesetzt werden!");
22         }
23
24         // 2. Abfrage(n) senden
25         $sql = 'SELECT * FROM country';
26         if ($ergebnis = $mysqli->query($sql)) {
27             // 3. Ergebniss(e) verarbeiten und ausgeben
28             echo "<h2>Es wurden " . $ergebnis->num_rows
29                 . " Länder gefunden.</h2>";
30
31             $s = "<table id=\"zebra\">\n\t";
32             $zaehler = 0;
33             while ($zeile = $ergebnis->fetch_object()) {

```

**Abb. 4.3:** Syntax-Highlighting im Notepad++

Editoren wie Notepad++ und Sublime Text sind keine Spezialisten für die Entwicklung von PHP-Anwendungen, sondern lediglich sehr gute und sehr leistungsfähige Editoren, mit denen Sie schon recht komfortabel codieren können.

Neben den genannten gibt es noch weitere Editoren, die von der Funktionsvielfalt alle recht ähnlich sind – zum Beispiel **Kate**, **Coda**, **Bluefish** oder **ActiveState Komodo Edit**.

Wenn die Möglichkeiten noch umfangreicher und komfortabler werden sollen, sollten Sie sich mit integrierten Entwicklungsumgebungen beschäftigen.

## 4.2 Integrierte Entwicklungsumgebung – IDE

Eine gute integrierte Entwicklungsumgebung (IDE = Integrated Development Environment) kann Ihnen die Entwicklungsarbeit erheblich erleichtern.

Die herausragenden Vorteile einer IDE liegen einerseits in der Integration verschiedener Werkzeuge und Funktionen unter einer gemeinsamen Oberfläche und andererseits in der Arbeit auf Projektebene.

Schauen wir uns an, was darunter zu verstehen ist.

### Projekt

Die Organisationseinheit innerhalb einer integrierten Entwicklungsumgebung ist das Projekt. Arbeiten auf Projektebene bedeutet hier: Anders als reine Editoren, die zwar einzelne Dateien bearbeiten, aber von anderen Dateien innerhalb desselben Projekts keine Kenntnis haben, wissen IDEs, welche Dateien zu einem Projekt gehören. Alles dreht sich also nicht um die Datei, sondern um das Projekt – bestehend aus (nahezu) beliebig vielen Dateien und Verzeichnissen. Dieser Ansatz wirkt sich zum Beispiel sehr vorteilhaft beim Editieren von Dateien (Auto-Completion, Code-Hinting) aus und ist auch beim Veröffentlichen oder beim Refaktorisieren<sup>23</sup> Ihrer Projekte hilfreich. Sie müssen sich an diesen Stellen nicht mehr komplett auf sich selbst verlassen, sondern erhalten Unterstützung durch die IDE.

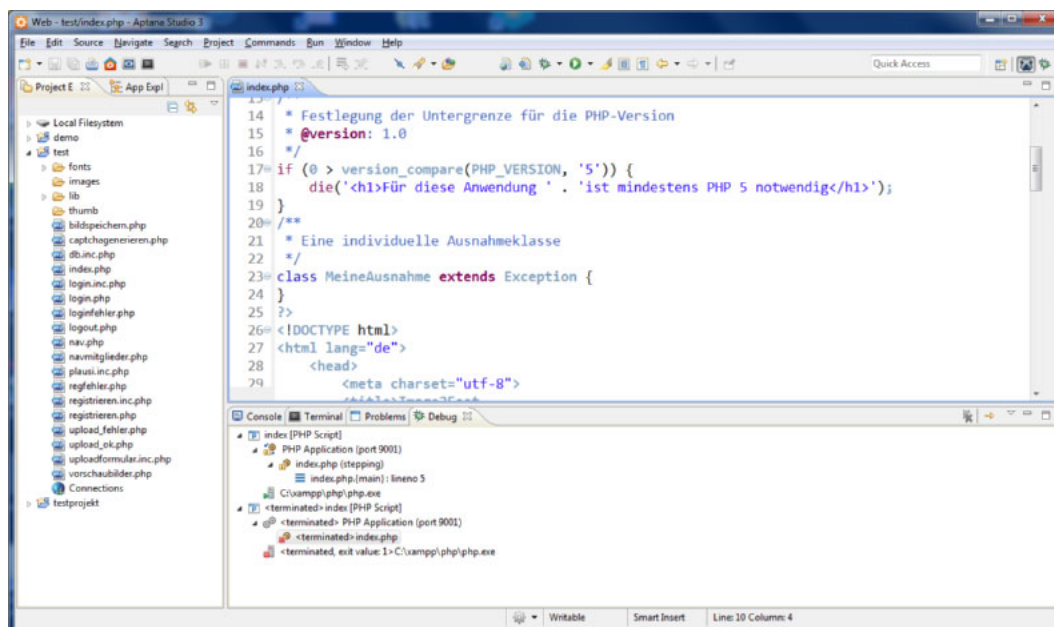


Abb. 4.4: Aptana Studio 3 mit geöffnetem Projekt

23. (engl. Refactoring) Die manuelle oder automatisierte Strukturverbesserung von Quellcode. Das primäre Ziel ist die Verbesserung des Codes hinsichtlich Lesbarkeit, Verständlichkeit, Wartbarkeit sowie Erweiterbarkeit, um den Aufwand beim Testen und der Weiterentwicklung zu reduzieren.

## Integration

Wie zuvor bereits erwähnt, ist unter Integration im Zusammenhang mit Entwicklungsumgebungen das Zusammenfassen von Werkzeugen und Funktionen unter einer gemeinsamen Oberfläche zu verstehen. Was bedeutet das konkret?

Bei der Entwicklung von Webanwendungen werden Sie immer wieder bestimmte Dinge erledigen müssen: neue Dateien erstellen, vorhandene kopieren, verschieben, löschen, editieren und so weiter. Gegebenenfalls und nicht unwahrscheinlich benötigt Ihr Projekt Datenbankunterstützung und es müssen hierfür administrative Dinge erledigt werden. Zum Beispiel müssen Sie Tabellen erstellen oder bearbeiten, Daten müssen importiert, geändert oder gelöscht werden und so weiter. Die Verbindungen zu Ihren Datenbanken und die Zugriffe müssen Sie selbstverständlich ebenfalls programmieren.

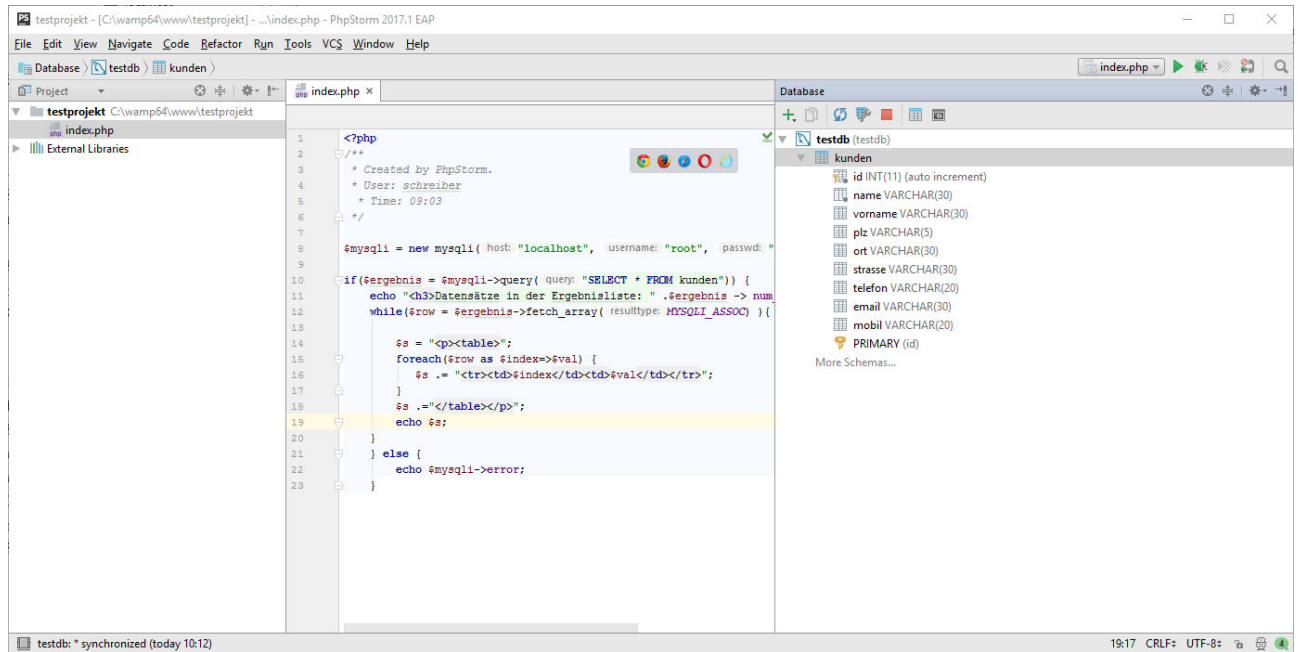
Gegebenenfalls möchten Sie in Ihren Projekten auch Versionsverwaltungssoftware wie Git einsetzen. Versionsverwaltungssoftware protokolliert unter anderem Änderungen an jeder einzelnen Datei über den Entwicklungszeitraum hinweg, sodass Sie Änderungen jederzeit nachvollziehen und auch auf frühere Entwicklungsstände zugreifen können. Diese Software müsste allerdings zunächst installiert und für die Zusammenarbeit mit Ihrem Projekt konfiguriert werden.

Diese Liste mit anfallenden Aufgaben und Wünschen ließe sich noch sehr lange fortsetzen. Sie möchten beispielsweise ein Framework einsetzen (wir werden gleich noch kurz anschauen, was das für PHP bedeutet).

Und auch bei der Fehlersuche und -diagnose möchten Sie vermutlich nicht allein gelassen werden und richten sich deshalb manuell einen sogenannten Debugger ein. Ein Debugger ist eine Software, die zur Laufzeit Ihre Anwendung an Haltepunkten unterbrechen und dann schrittweise bestimmte Passagen durchlaufen kann. Währenddessen können Sie sich die Werte und Zustände von Variablen und Objekten anzeigen lassen.

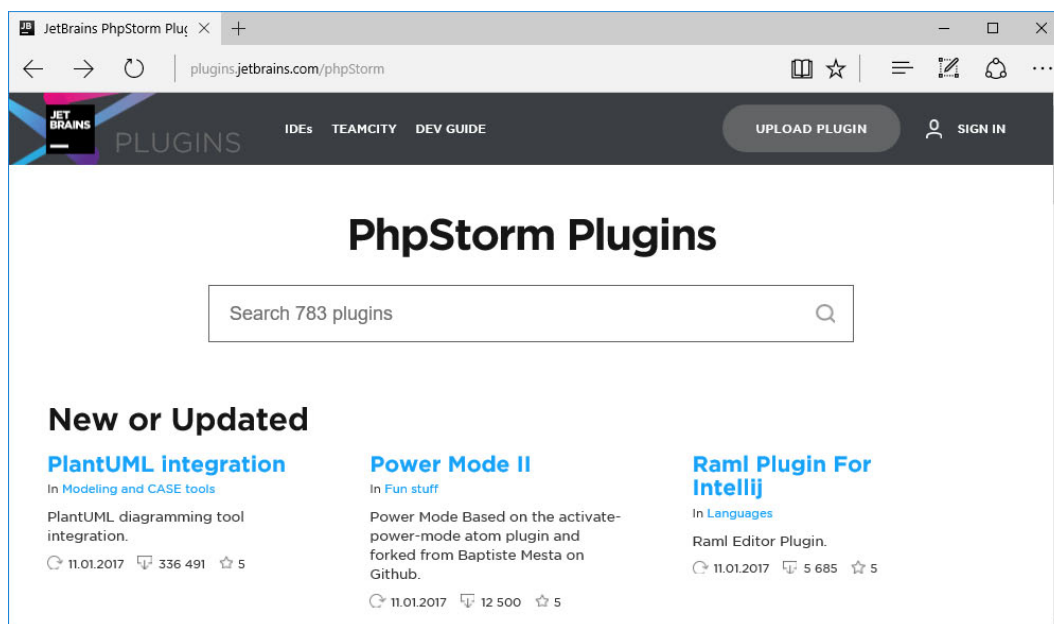
Das könnten Sie jetzt alles einzeln mit verschiedenen Entwicklungstools für Ihr Projekt machen. Und manches für das nächste Projekt dann wieder und wieder.

Viel besser, einfacher, schneller und effizienter erledigen Sie das aber, indem Sie zu einer integrierten Entwicklungsumgebung greifen. Diese stellt schon einige der genannten Dinge in der Standardinstallation zur Verfügung. Die Projektverwaltung und ein in der Regel guter Editor sind auf jeden Fall dabei und direkt einsatzbereit. Vieles andere lässt sich sehr leicht hinzufügen.



**Abb. 4.5:** PhpStorm mit Projektverwaltung, Editor und Datenbankansicht (von links)

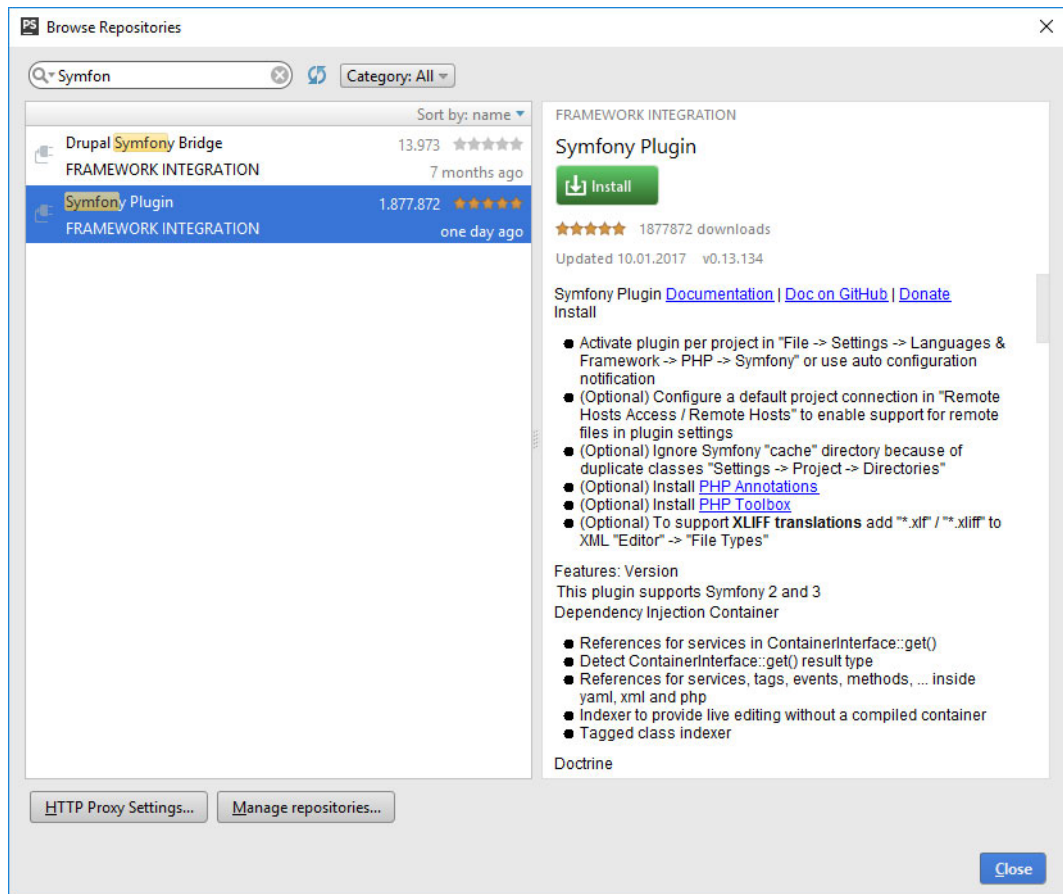
Gute IDEs lassen sich sinnvoll in ihrem Funktionsumfang erweitern. Über Erweiterungen – sogenannte Plugins – können Sie so die IDE exakt an Ihre Bedürfnisse anpassen. Welche Plugins überhaupt für eine IDE zur Verfügung stehen, ist von IDE zu IDE stark schwankend und erheblich davon abhängig, wie populär beziehungsweise marktrelevant die Entwicklungsumgebung ist. Für das zuvor abgebildete PhpStorm des Unternehmens JetBrains stehen derzeit mehr als 750 Plugins zum Download zur Verfügung. PhpStorm hat eine ständig wachsende Anhängerschaft und ist bei PHP-Entwicklern sehr beliebt.



**Abb. 4.6:** Verfügbare Plugins für PhpStorm



Noch komfortabler als der Download von der JetBrains-Website ist die Installation benötigter Plugins direkt aus der IDE selbst. Sie müssen also nicht einmal einen Browser öffnen, um ein fehlendes Plugin zu suchen und zu installieren.



**Abb. 4.7:** Symfony Plugin zur Framework-Integration direkt aus PhpStorm installieren

Und auch der zuvor bereits kurz erwähnte Debugger ist in einer IDE – mehr oder weniger schnell und komplikationslos eingerichtet – vorhanden und verrichtet dort dann zuverlässig und dauerhaft seinen Dienst.



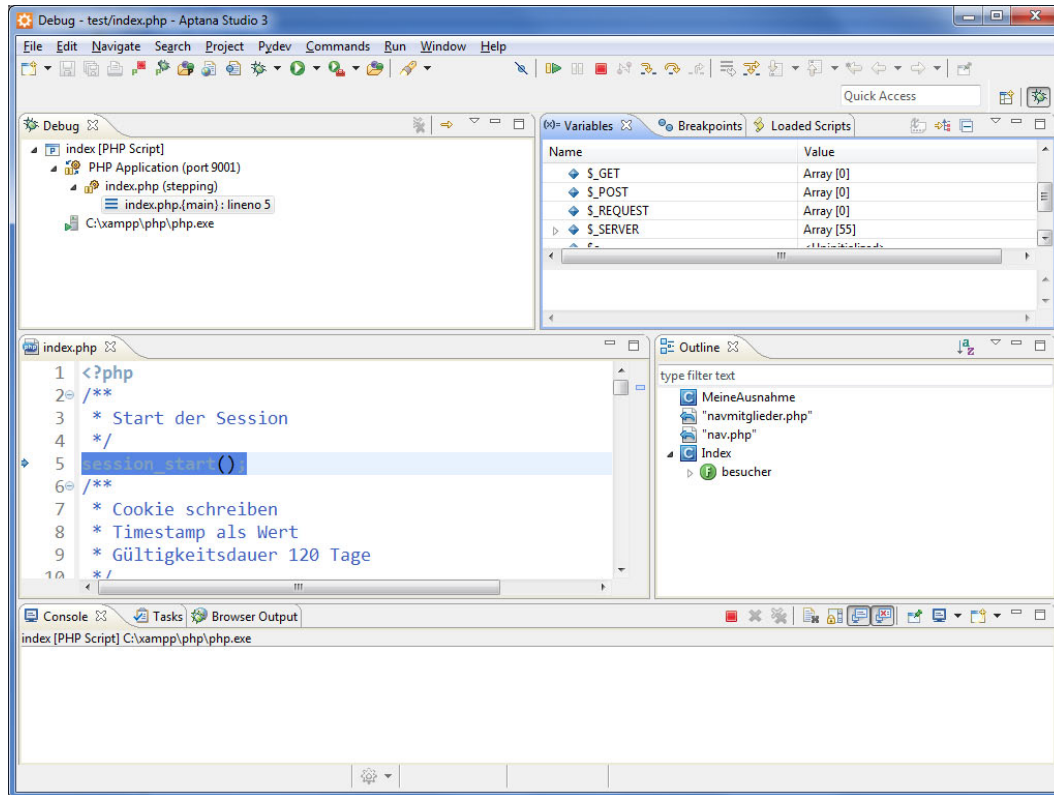


Abb. 4.8: Debugging in Aptana Studio 3

Die Auswahl der richtigen IDE zur Entwicklung von Webanwendungen ist nicht ganz einfach. Es gibt zahlreiche<sup>24</sup> und diese unterscheiden sich alle irgendwie, aber manche auch nur im Detail. Einige stammen von anderen ab, sind Abspaltungen desselben Projekts oder haben zumindest dieselben Wurzeln. Manche können nur mit wenigen Programmiersprachen umgehen und andere dagegen mit vielen.

Ein Kriterium für die Auswahl der IDE könnte der Preis sein. Entscheiden Sie zwischen kostenlosen und kostenpflichtigen Entwicklungsumgebungen. Beispiele für kostenlose IDEs sind das zuvor auch bereits mehrfach abgebildete **Aptana Studio**, **Eclipse PDT** oder **Netbeans**. Kostenpflichtig sind zum Beispiel **PhpStorm** oder **Zend Studio**.

Gerade beim Reinschnuppert ist der Preis sicher ein naheliegendes und gerne verwendetes Auswahlkriterium. Beachten Sie aber, dass die kostenpflichtigen IDEs häufig interessante Lizenzierungsmodelle anbieten, oft auch gute Produktunterstützung liefern und manches im Detail vielleicht etwas besser lösen.

In jedem Fall sollten Sie mehrere IDEs testen, bevor Sie sich auf eine festlegen. Download und Installation sind meist schnell erledigt. Die freien IDEs sind sowieso verfügbar und für die kostenpflichtigen stehen meistens kostenlose Testzeiträume zur Verfügung.

Schauen Sie nach, ob die IDE sich für Sie gut bedienen und an Ihre individuellen Bedürfnisse anpassen lässt. Was lässt sich wie einstellen? Sind Tastenkombinationen editierbar, was ist mit Schriften, Farben oder Themes? Stellt die IDE ausreichende Hilfefunktionen zur Verfügung, gibt es eine großes Community und wie ist der Support des Herstellers?

24. Für einen Überblick finden Sie eine Liste der PHP-Editoren und IDEs unter [https://en.wikipedia.org/wiki/List\\_of\\_PHP\\_editors](https://en.wikipedia.org/wiki/List_of_PHP_editors) (englisch)

Selbstverständlich müssen Sie beachten, ob die IDE auch für die Plattform verfügbar ist, auf der Sie entwickeln wollen. Die bedeutendsten IDEs stehen allerdings für Windows, Mac und Linux zur Verfügung.

Mit zunehmender Professionalisierung der Entwicklertätigkeit wird auch die Erweiterbarkeit und der Funktionsumfang der IDE zunehmend relevant. Was lässt sich an Funktionen integrieren, die für Ihre späteren Projekte benötigt werden? Und dabei geht es nicht immer nur darum, ob eine bestimmte Funktion überhaupt zur Verfügung steht, sondern wie komfortabel, umfangreich oder performant diese ist.

Für den interessierten Einsteiger im Bereich PHP-Anwendungsentwicklung sind die Fragen zur Erweiterbarkeit voraussichtlich am schwierigsten für sich zu beantworten, da in diesem Punkt häufig noch der Überblick über die Möglichkeiten fehlt.

Es bestehen zudem zahlreiche weitere Kriterien für die Auswahl der geeigneten IDE. Für unseren kurzen Überblick soll das aber ausreichen.



Probieren Sie die Entwicklungsumgebungen aus, die für Sie infrage kommen. Erfahrungsgemäß merken Sie recht schnell, ob eine IDE für Sie geeignet ist oder nicht.

### 4.3 PHP-Frameworks

Zum Schluss dieser Lektion schauen wir uns noch kurz gemeinsam an, was PHP-Frameworks sind, welchen Zweck sie haben und wie Sie ein geeignetes Framework auswählen.

An die Antwort auf die Frage, was ein Framework überhaupt ist, können Sie sich auf verschiedene Arten annähern, denn ein Framework vereint Konzepte, Bibliotheken und Funktionen unter einer gemeinsamen Bezeichnung.

Verstehen Sie – für das Verständnis in diesem Studienheft – ein PHP-Framework der Einfachheit halber als eine Art Rahmen oder Gerüst, der beziehungsweise das nach einem bestimmten Architekturmuster funktioniert, mit dem Ziel, Sie bei der Entwicklung Ihrer Webanwendungen vielfältig zu unterstützen. Mit einem Framework werden Sie effizienter entwickeln, besseren Code schreiben, weniger Fehler produzieren und sicherere Anwendungen erstellen.

PHP-Frameworks haben bereits Lösungen für sehr viele Programmierprobleme standardmäßig an Bord, die in Webanwendungen immer wieder auftauchen. Sie müssen das Rad nicht immer neu erfinden, sondern sollten auf fertige, funktionierende und sichere Lösungen zurückgreifen.

Wesentliche Vorteile beim Einsatz eines guten PHP-Frameworks sind unter anderem:

- Organisation der Dateien und Verzeichnisse
- Vorgefertigte Codebibliotheken
  - Datenbankabstraktion
  - Formularvalidierung
  - Ein- und Ausgabefilter
  - Sitzungsverwaltung
  - Sicherheitsaspekte (XSS, SQL-Injection etc.)
  - ...

- Schnelle Anwendungsentwicklung (Rapid Application Development)
- Entwicklung im Team
- Unterstützung durch die Framework-Community
- u. v. m.

Wir hatten bei den ersten Schritten in PHP zuvor einmal als Vorteil genannt, dass sich PHP-Code einfach und beliebig oft in HTML einbetten lässt. Das ist für den Einstieg auch sicher ein Vorteil, denn Sie können direkt starten. Für die Übersichtlichkeit, Wartbarkeit, Wiederverwendbarkeit des Codes und für die Arbeit im Entwicklerteam ist das aber genau das Gegenteil eines Vorteils. Diese Art der Programmierung hat im professionellen Bereich gravierende Nachteile.

Viele Frameworks – nicht nur für die Entwicklung mit PHP – basieren deshalb auf dem MVC-Architekturmuster, bei dem eine Anwendungskomponente in drei Module gegliedert wird: in das **Model**, die **View** und den **Controller**. Das sind eigenständige Module, die jeweils Teilaufgaben übernehmen. Auch wenn wir hier auf die Details dieser Architektur verzichten möchten, schauen wir uns kurz an, wofür die drei Teile stehen.

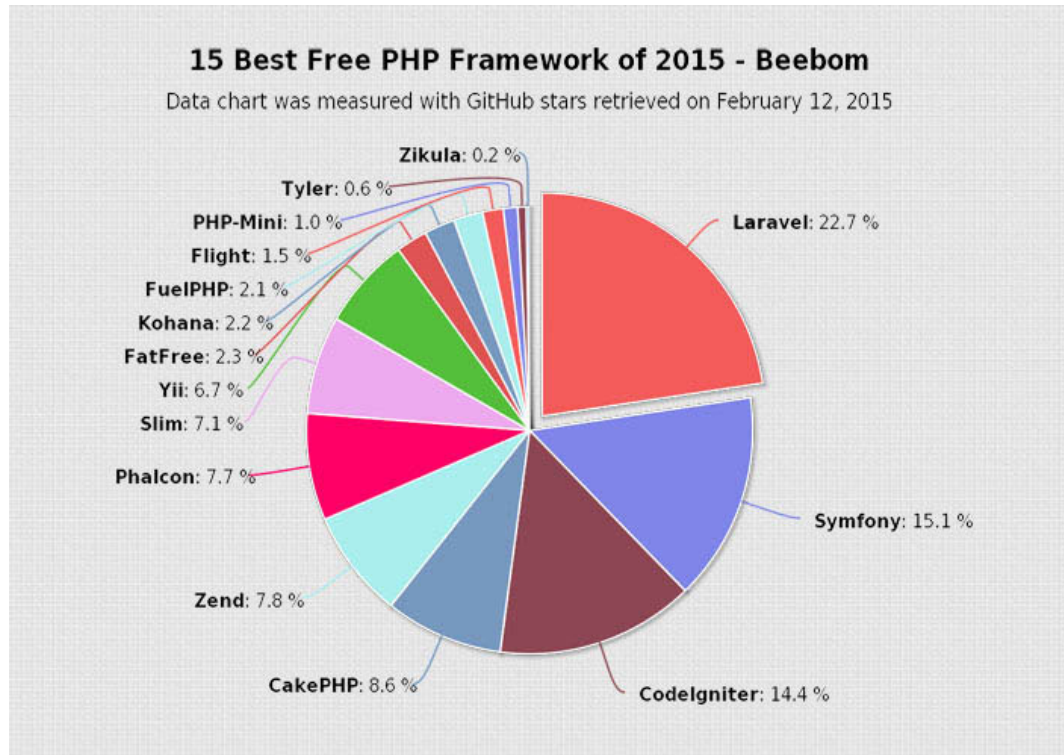
Das **Model** (dt. Modell) repräsentiert die Daten. Es hat die primäre Aufgabe, die Anwendungskomponente mit Daten zu versorgen. Auch die Datenspeicherung und die Kommunikation mit der oder den Datenquellen obliegt dem Model.

Die Darstellung der Daten erfolgt über die **View** (dt. Präsentation). Häufig werden in diesem Bereich entsprechend mächtige **Template Engines** eingesetzt, mit denen die Ausgaben und Formulare effizient und einheitlich in hoher Qualität erzeugt werden können.

Die (Geschäfts-)Logik der Anwendungskomponente befindet sich schließlich im **Controller** (dt. Steuerung).

Die beschriebene Dreiteilung und damit die Trennung von Daten, Logik und äußerem Erscheinungsbild hat erhebliche Vorteile bei Änderungen und Erweiterungen der Anwendung.

Das Angebot an PHP-Frameworks ist riesig und selbst das der guten Frameworks ist groß. Im Internet gibt es immer wieder umfangreiche Listen und Rankings der besten beziehungsweise populärsten Frameworks. Wie das mit den Informationen aus dem Internet ist, sind diese häufig weder objektiv noch repräsentativ. Allerdings lassen sich recht gut Tendenzen erkennen, welche Frameworks in der jüngeren Zeit besonders marktrelevant waren. Wenn Sie dann noch ein wenig auf die Quelle achten, werden Sie bemerken, dass sich unter den Top-Empfehlungen immer dieselben Kandidaten befinden.



**Abb. 4.9:** Die 15 „besten“ kostenlosen PHP-Frameworks im Jahr 2015  
(Quelle: <http://beebom.com/>)

Die am häufigsten genannten PHP-Frameworks sind regelmäßig:

- Laravel,
- Symfony,
- CodeIgniter,
- CakePHP,
- Zend,
- Phalcon,
- Slim und
- Yii.

Die Namen blieben in den letzten Jahren dieselben. Lediglich die Positionen unter den Top 5 oder 10 wurde an der einen oder anderen Stelle schon einmal getauscht. Die letzte richtige Neuentdeckung unter den relevanten PHP-Frameworks war Laravel. Es handelt sich dabei um ein noch recht junges Framework aus dem Jahr 2012, das sich sehr schnell in der Spitze der Popularitätsliste etabliert hat und sie derzeit auch anführt.

## Das richtige Framework

Die Auswahl des richtigen PHP-Frameworks ist nicht ganz einfach und immer auch etwas mühselig. In der Regel es ist nicht sinnvoll, ein Framework zu häufig zu wechseln. Die Einarbeitung in ein Framework dauert einige Zeit und ist bis zu dem Moment, an dem Sie von den Vorteilen so richtig profitieren können, mit einem gewissen Aufwand verbunden. Die Entscheidung für oder gegen ein bestimmtes Framework ist also eher eine **strategische Entscheidung** als eine für das nächste anstehende Projekt.

Schauen wir uns flüchtig einige Kriterien an, die Sie bei der Auswahl eines PHP-Frameworks berücksichtigen können. Es gibt weitere, die hier nicht genannt werden.

Wichtig ist selbstverständlich der **Funktionsumfang**. Nur wenn ein Framework funktional die Anforderungen erfüllt, kann es in die engere Wahl kommen. Außerdem sollten Sie im Auge behalten, ob das Framework **technisch noch auf dem aktuellen Stand** ist. Werden neue Versionen unterstützt und wie aktiv wird das Framework weiterentwickelt? Wie ist gegebenenfalls die **Performance** des Frameworks? Gilt es als schnell oder als langsam?

Gerade für den Einstieg in ein Framework ist es von erheblichem Belang, wie leicht dieser Einstieg fällt. Es gibt Frameworks, die als besonders **einsteigerfreundlich** gelten, und andere, deren Konzepte einen hohen **Eingewöhnungsaufwand** erfordern. Ganz wichtig sind in diesem Zusammenhang auch der Umfang und die **Qualität der Dokumentation** des Frameworks. In diesen Kontext fallen auch die Punkte **Verbreitung** und **Community**. Je weiter ein Framework verbreitet ist und je aktiver die Community, umso leichter wird es Ihnen fallen, im Bedarfsfall Antworten auf Fragen zum Framework zu erhalten.

Nicht außer Acht lassen sollten Sie auch die Entwickler des Frameworks. Handelt es sich dabei um mehrere Personen und ist der Fortbestand des Frameworks wahrscheinlich? Wie ist die **Qualität** des erzeugten Codes und wie wird mit **Fehlern** im Framework umgegangen? Wie sieht es mit dem **Qualitätsmanagement** generell aus?

Außerdem ist es auch wichtig, unter welcher **Lizenz** das Framework veröffentlicht wurde.

Nicht zuletzt sollten Sie sich auch ein wenig auf Ihr Gespür verlassen. Geht Ihnen das Arbeiten mit dem Framework leicht von der Hand und macht es Spaß, damit zu arbeiten? Das ist ein Kriterium, das für den Entwicklungsalltag besonders wichtig ist.

So viel zu Editoren, Entwicklungsumgebungen und Frameworks für PHP. Fassen wir das wichtigste noch einmal zusammen.

## Zusammenfassung

Für die Entwicklung von PHP-Anwendungen stehen viele Editoren und IDEs zur Verfügung. Gute Editoren unterstützen Sie mit Funktionen wie Syntax-Highlighting, Auto-Completion und Code-Hinting.

Integrierte Entwicklungsumgebungen arbeiten auf Projektebene und integrieren viele Funktionen unter einer einheitlichen Oberfläche. Sie lassen sich häufig stark individualisieren und über Plugins erweitern.

PHP Frameworks bieten viele Features, um Webanwendungen schnell und in guter Qualität zu entwickeln. Die Auswahl eines geeigneten Frameworks und die Einarbeitung in das gewählte Framework können unter Umständen recht aufwendig sein.

## Aufgaben zur Selbstüberprüfung

- 4.1 Nennen Sie zwei Editoren und zwei IDEs für die PHP-Entwicklung.
- 4.2 Was ist Syntax-Highlighting?
- 4.3 Was ist ein Debugger?

## 5 XML

*In der letzten Lektion dieses Studienhefts beschäftigen wir uns noch etwas mit XML.*

*Sie erhalten einen kurzen Überblick darüber, was XML ist und wie eine XML-Datei aufgebaut ist.*

*Ganz zum Schluss schlagen wir dann in einem kleinen Beispiel noch einmal den Bogen zu serverseitiger Programmierung mit PHP und schauen uns an, wie Sie mit einer PHP-Standarderweiterung eine XML-Datei auslesen und mit den Daten im Skript arbeiten.*

Schauen wir uns zunächst an, wofür die Abkürzung **XML** steht und was dieses XML überhaupt ist.

### 5.1 Was ist XML?

**XML** steht für **eXtensible Markup Language** (dt. erweiterbare Auszeichnungssprache). Hervorgegangen ist XML aus SGML (Standard Generalized Markup Language). Genauer gesagt ist XML eine Untermenge von SGML. XML wurde in der Version 1.0 im Februar 1998 durch das World Wide Web Konsortium (W3C) als Standard zur Dokumentenauszeichnung eingeführt. XML dient dazu, Daten in strukturierter Form als Text zu speichern.

XML ist in der Informationstechnologie auch fast 20 Jahre nach Veröffentlichung noch von erheblicher Bedeutung. Kaum ein IT-Bereich kommt heutzutage gänzlich ohne XML-Berührung aus. Auch in der Softwareentwicklung gibt es, unabhängig von Programmiersprache und Programmierparadigma, ständig Kontakt zu XML.

XML ist ein offener Standard, lizenzfrei und kann beliebig in einer Anwendung eingesetzt werden. Da die Daten im Textformat abgespeichert werden und es für alle Plattformen „XML-fähige“ Anwendungen gibt, ist auch XML selbst plattformunabhängig.

Die Speicherung als Text hat den Vorteil, dass XML-Dokumente vielseitig verwendbar und höchstwahrscheinlich auch in weiter Zukunft noch lesbar sind.

Der Nachteil der größeren Datenmenge bei der Speicherung in Textform verlor zusehends an Bedeutung, da Speicherplatz in immer größerer Menge zu immer geringeren Kosten zur Verfügung stand.

### 5.2 Aufbau eines XML-Dokuments

Ob ein Textdokument ein XML-Dokument ist, hängt vor allem davon ab, ob es den syntaktischen Vorgaben des XML-Standards folgt. Im Fachjargon spricht man von „wohlgeformt“. Schauen wir uns den Aufbau und einige Regeln an.

Ein XML-Dokument ist hierarchisch aufgebaut. Es besteht aus einem Dokumentkopf, dem sogenannten **Prolog**, und den eigentlichen **Daten**. Sie kennen das hierarchische Prinzip grundsätzlich bereits von HTML und werden sicher Parallelen erkennen.



Ein Beispiel für eine XML-Datei:

```
<?xml version="1.0" encoding="UTF-8"?>
<Rechnung>
  <RNR>100</RNR>
  <Datum>2016-02-01</Datum>
  <Kunde>Huber</Kunde>
  <Ort>Hamburg</Ort>
  <Strasse>Lange Straße 23</Strasse>
  <Positionen>
    <Position>
      <Artikel>Fahrrad</Artikel>
      <Preis>399</Preis>
      <Anzahl>2</Anzahl>
    </Position>
    <Position>
      <Artikel>Helm</Artikel>
      <Preis>39.5</Preis>
      <Anzahl>2</Anzahl>
    </Position>
    <Position>
      <Artikel>Kette</Artikel>
      <Preis>99.9</Preis>
      <Anzahl>2</Anzahl>
    </Position>
  </Positionen>
</Rechnung>
```

**Code 5.1:** Beispiel einer einfachen XML-Datei (*rechnung.xml*)

In dieser XML-Datei sind einige Daten einer Rechnung gespeichert.

### 5.2.1 Prolog

Zu Beginn einer XML-Datei steht der sogenannte Prolog. Der Prolog ist zwar optional, doch kennzeichnet er die Datei direkt als XML-Dokument. Es ist deshalb sinnvoll und ausdrücklich empfohlen, einen Prolog zu notieren.

Zu Beginn des Prologs steht die sogenannte XML-Deklaration.

```
<?xml version="1.0" encoding="UTF-8"?>
```

Die XML-Deklaration enthält immer Angaben zur verwendeten XML-Version. Das ist in der Regel die Version 1.0.



Die XML-Deklaration muss direkt zu Beginn der Datei stehen. Es dürfen weder Leerzeilen noch Leerzeichen davor stehen.

Neben dem Attribut `version` kann die XML-Deklaration noch weitere Attribute umfassen. Im Beispiel zuvor ist das auch noch das Attribut `encoding`. Als Wert dieses Attributs wird die Zeichensatzcodierung angegeben. Hier ist das die Codierung UTF-8 des Zeichensatzes Unicode. UTF-8 ist auch die XML-Standard-Codierung.



Manchmal finden Sie in der XML-Deklaration auch noch das Attribut `standalone`. Als Attributwerte stehen für dieses Attribut nur die Werte `"yes"` oder `"no"` zur Verfügung. Der Standardwert ist `"no"` (er kann deshalb weggelassen werden) und bedeutet, dass für dieses Dokument nach Dokumenttypdefinitionen geschaut wird.

Die Dokumenttypdefinition ist die Grammatik für die spezielle XML-Sprache, die in dieser Anwendung benutzt wird. Die DTD listet alle Elemente und Attribute auf, die im Dokument benutzt werden können. Die DTD kann in der XML-Datei selbst oder in einer externen DTD-Datei stehen.

Die Attribute `encoding` und `standalone` sind optional. Sollten beide angegeben sein, müssen sie zwingend in der Reihenfolge `encoding` und dann `standalone` stehen.

Der Prolog einer XML-Datei kann noch weitere Angaben enthalten. Häufig finden Sie noch eine Dokumenttypdeklaration. Sie kennen das bereits aus HTML. Für HTML 5 war das zum Beispiel

```
<!DOCTYPE html>
```

oder für XHTML Transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Die Dokumenttypdeklaration stellt dabei den Bezug zur Dokumenttypdefinition her. Wenn für eine angegebene Dokumenttypdeklaration eine gültige Dokumenttypdefinition vorliegt, gilt das XML-Dokument als **gültig**. Gültigkeit ist neben der Wohlgeformtheit des XML-Codes das zweite Kriterium für ein korrektes XML-Dokument.

So viel zum Prolog einer XML-Datei. Sehen wir uns den restlichen Aufbau jetzt auch noch an.

## 5.2.2 XML-Elementbaum

Auf den Prolog folgen in einer hierarchischen Struktur aus Elementen die eigentlichen XML-Daten. Ganz oben in der Hierarchie befindet sich das **Wurzelement** – auch **Root-Element** oder **Stammelement** genannt. Alle weiteren Elemente müssen sich innerhalb dieses einen Elements befinden. In unserer Beispieldatei von zuvor ist das Element `<Rechnung>` das Root-Element.

```
<?xml version="1.0" encoding="UTF-8"?>
<Rechnung>
...
</Rechnung>
```

Alle weiteren Elemente sind in einer Baumstruktur ineinander verschachtelt. Jedes Element hat genau ein Elternelement<sup>25</sup> und kann weitere Kindelemente enthalten.

Da Ihnen diese Art des Elementbaums mit seinen Verschachtelungen aus HTML bereits sehr gut bekannt ist, gehen wir hier nicht näher darauf ein.

25. Außer das Root-Element. Es hat kein Elternelement, da es ganz oben in der Hierarchie steht.

Ebenfalls wie in HTML besteht ein Element aus einem **Start**-Tag und einem **End**-Tag, beides jeweils in spitzen Klammern notiert. Das End-Tag enthält zusätzlich einen Schrägstrich (/) zu Beginn.

Zwischen Start- und End-Tag stehen entweder weitere Elemente, Content oder beides. Elemente ohne Inhalt werden als „leere Elemente“ bezeichnet und dürfen als einzelnes Tag notiert werden – `<leeresElement />`.

Anders als in HTML können die Namen der Elemente – was die Anforderungen an die Wohlgeformtheit betrifft – frei vergeben werden. Ein Element muss dabei entweder mit einem Buchstaben, einem Unterstrich oder einem Doppelpunkt beginnen. Es gibt einige Konventionen und Empfehlungen für die Namensvergabe, die wir uns hier im Detail nicht ansehen. Für alphanumerische Zeichen gibt es keine Probleme.

Für die Elementnamen gilt, dass zwischen Groß- und Kleinschreibung **unterschieden wird**. Das Element `<Element></Element>` ist ein vollständig anderes als das Element `<element></element>`.

Die Elemente können mit Attributen versehen werden. Jedes Attribut besteht dabei aus einem Attributnamen und dem Attributwert. Attribute werden eingesetzt, um Eigenschaften eines Elements zu beschreiben. Attribute werden immer im Start-Tag notiert. Mehrere Attribute werden durch ein Leerzeichen voneinander getrennt. Ein bestimmtes Attribut darf es für ein Element nur einmal geben.

```
<ball farbe="rot" durchmesser="25" material="gummi">Kinderball
</ball>
```

Im Beispiel oben hat das Element `ball` die Attribute `farbe`, `durchmesser` und `material` mit entsprechenden Attributwerten.

Von der Definition her ist ein Attribut etwas, das das eigentliche Element näher beschreibt – also in gewisser Weise Zusatzinformationen ähnlich einer Eigenschaft zum Element zur Verfügung stellt. In der Praxis ist es aber nicht immer deutlich trennbar, ob etwas selbst ein Element ist oder ob es Attribut eines anderen Elements ist. Das Beispiel von zuvor mit dem Ball könnte auch so notiert werden:

```
<ball name="kinderball">
  <farbe>rot</farbe>
  <durchmesser>25</durchmesser>
  <material>gummi</material>
</ball>
```

Es würden im Wesentlichen dieselben Informationen gespeichert.

So viel an dieser Stelle zu XML-Dokumenten und deren Aufbau. Schauen wir uns zum Schluss dieses Studienhefts noch kurz an einem kleinen Beispiel an, wie PHP mit einem XML-Dokument umgeht.

### 5.3 XML-Dateien und PHP

#### Hinweis:

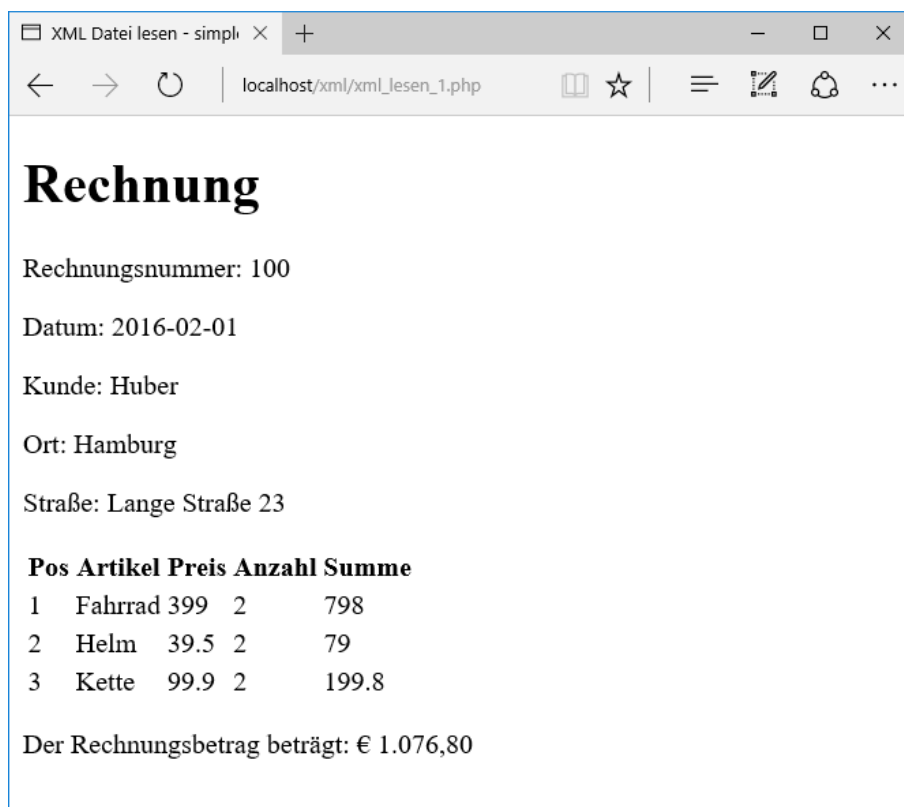
PHP stellt mittlerweile in einer Standardinstallation mehrere Erweiterungen zum Arbeiten mit XML-Dokumenten zur Verfügung. Wir werden uns hier beispielhaft den lesenden Zugriff mit der Erweiterung **SimpleXML** ansehen.

Selbstverständlich müssen Sie das folgende PHP-Codebeispiel nicht im Detail nachvollziehen können. Es dient nur zur Demonstration, wie einfach die Zusammenarbeit ist, wenn die Datenquelle eine XML-Datei ist.

Nehmen wir als Ausgangspunkt noch einmal unsere XML-Datei *rechnung.xml*, wie sie unter Code 5.1 angegeben ist.

Wir möchten diese Datei nun auslesen, zusätzlich einige einfache Summen berechnen und die Daten und Ergebnisse im Browser ausgeben. Wir erstellen dazu ein einfaches PHP-Skript und speichern dieses zusammen mit der XML-Datei im selben Unterverzeichnis. Dieses Verzeichnis liegt unterhalb des Document-Root-Verzeichnisses des Webserver, damit das Skript auch ausgeführt werden. Sie erinnern sich, dass das so sein muss.

Das Skript erzeugt folgende Ausgabe:



**Abb. 5.1:** Ausgabe im Browser

Sie sehen, dass die Daten aus dem XML-Dokument in die Ausgaben übernommen wurden. Zusätzlich haben wir noch nummerierte Positionen, Summen für die Positionen und ganz unten die Ausgabe des berechneten Rechnungsbetrages.

Hier ist zunächst das komplette PHP-Skript. Wir werden es dann abschnittsweise im Detail ansehen.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>XML Datei lesen - simplexml_load_file</title>
</head>
<body>
<?php

//falls die Datei vorhanden ist
if (file_exists("rechnung.xml")) {
  //Objekt erstellen
  //Enthält die Daten und die Struktur aus der
  //XML-Datei
  $xml = simplexml_load_file("rechnung.xml");
} else {
  //Falls das Öffnen scheitert
  exit("Die Datei rechnung.xml ist nicht vorhanden oder konnte
  nicht geöffnet werden.");
}

//Die Ausgabe in einer Zeichenkettenvariablen zusammenstellen
//.= hängt an die bereits bestehende Zeichenkette an.

$ausgabe = "<h1>Rechnung</h1>\n";
$ausgabe .= "<p>Rechnungsnummer: " . $xml->RNr . "</p>\n";
$ausgabe .= "<p>Datum: " . $xml->Datum . "</p>\n";
$ausgabe .= "<p>Kunde: " . $xml->Kunde . "</p>\n";
$ausgabe .= "<p>Ort: " . $xml->Ort . "</p>\n";
$ausgabe .= "<p>Straße: " . $xml->Strasse . "</p>\n";
$pos = 0;
$gesamtpreis = 0.0;

//Beginn einer Tabelle erstellen und dem Ausgabestring anhängen
$ausgabe .= "<table>\n\t<tr>\n\t\t<th>Pos</th><th>Artikel</th><th>Preis</th><th>Anzahl</th><th>Summe</th>\n\t</tr>\n\t<tbody>\n\t\t";
//Positionen in einer foreach-Schleife durchlaufen
//und dabei Tabellenzeilen generieren
foreach($xml->Positionen->children() as $position) {
  $summepos = floatval($position->Preis) * $position->Anzahl;
  $gesamtpreis += $summepos;
  $ausgabe .= "<tr>\n\t\t<td>" . (++$pos) . "</td>";
  $ausgabe .= "<td>". $position->Artikel . "</td>";
  $ausgabe .= "<td>". $position->Preis . "</td>";
  $ausgabe .= "<td>". $position->Anzahl . "</td>";
  $ausgabe .= "<td>". $summepos . "</td>\n\t</tr>\n\t";
}

$ausgabe .= "</tbody>\n</table>\n";
```

```
//Ausgabestring abschließen
$ausgabe .= "<p>Der Rechnungsbetrag beträgt: € "
            .(number_format($gesamtpreis, 2, ",", ".")) . "</p>\n";

//ausgeben
echo $ausgabe;
?>
</body>
</html>
```

**Code 5.2:** Eine XML-Datei mit PHP lesen

Zu Beginn des PHP-Bereichs überprüfen wir, ob die XML-Datei, die wir lesen möchten, überhaupt vorhanden ist. Wenn das der Fall ist, dann verwenden wir die Funktion `simplexml_load_file()`, um ein Objekt `$xml` zu erstellen, das die Daten aus der XML-Datei enthält.

```
if (file_exists("rechnung.xml")) {
    $xml = simplexml_load_file("rechnung.xml");
} else {
    exit("Die Datei rechnung.xml ist nicht vorhanden oder konnte
    nicht geöffnet werden.");
}
```

Für den Fall, dass das nicht klappt, geben wir eine Meldung aus und beenden das Skript.

Im besten Fall stehen uns jetzt schon alle Informationen aus der XML-Datei über die Objektvariable `$xml` in unserem Skript zur Verfügung. Jetzt können wir damit arbeiten.

Im ersten Schritt bereiten wir ein paar Ausgaben vor. Anstatt alles einzeln mit dem Befehl `echo` auszugeben, haben wir hier die Variante gewählt, dass wir zunächst die komplette Ausgabe in einer Zeichenkettenvariablen `$ausgabe` zusammensetzen. Das Verketteten erfolgt dabei mit dem Punkt-Operator (`.`). Der kombinierte Operator `.=` hängt die zu verkettenden Zeichen an die bereits bestehende Zeichenkette an.

```
$ausgabe = "<h1>Rechnung</h1>\n";
$ausgabe .= "<p>Rechnungsnummer: " . $xml->RNR . "</p>\n";
$ausgabe .= "<p>Datum: " . $xml->Datum . "</p>\n";
$ausgabe .= "<p>Kunde: " . $xml->Kunde . "</p>\n";
$ausgabe .= "<p>Ort: " . $xml->Ort . "</p>\n";
$ausgabe .= "<p>Straße: " . $xml->Strasse . "</p>\n";
```

Der Zugriff auf die Elemente der XML-Datei, von denen jetzt ja eine Kopie in unserem Objekt `$xml` vorliegt, erfolgt wie bei Objekten unter PHP üblich mit dem Operator `->`. Als Bezeichner verwenden Sie den Elementnamen aus der XML-Datei.

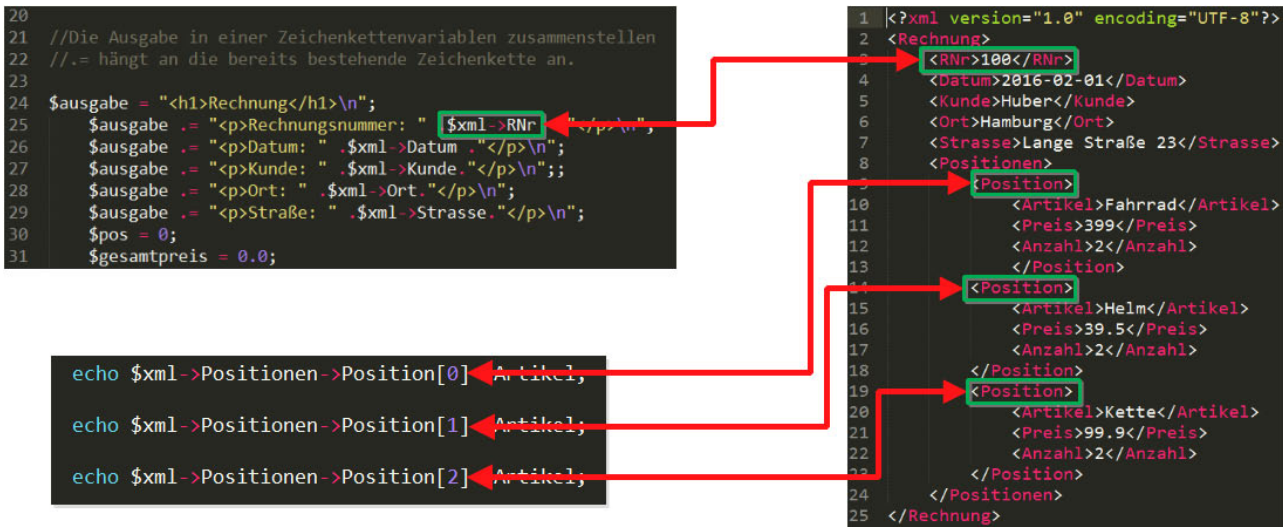


Abb. 5.2: Zugriff auf die Elemente

Auf Elemente mit demselben Elementnamen, die sich auf derselben Hierarchieebene befinden, können Sie über einen Index zu greifen. Dieser Index ist nullbasiert. Das bedeutet, das erste Element sprechen Sie über den Index 0 und das letzte über den Index Anzahl – 1 an. Beispiele dafür sehen Sie in der Abb. 5.2 unten links. Hier wird für die drei Positionen (am Index 0, 1 und 2) jeweils auf das Element Artikel zugegriffen.

Wenn Sie auf alle Elemente einer solchen Gruppe von Elementen zugreifen möchten, ist es oft am besten, diese in einer Schleife abzuarbeiten. Sie codieren einen Sachverhalt dann nur einmal, unabhängig davon, wie viele solcher Elemente vorhanden sind. Genauso haben wir das auch in unserem Beispiel-Skript von oben getan.

```

foreach($xml->Positionen->children() as $position) {
  $summepos = floatval($position->Preis) * $position->Anzahl;
  $gesamtpreis += $summepos;
  $ausgabe .= "<tr>\n\t\t<td>" . (++$pos) . "</td>";
  $ausgabe .= "<td>". $position->Artikel . "</td>";
  $ausgabe .= "<td>". $position->Preis . "</td>";
  $ausgabe .= "<td>". $position->Anzahl . "</td>";
  $ausgabe .= "<td>". $summepos . "</td>\n\t</tr>\n\t";
}

```

In einer `foreach`-Schleife durchlaufen wir nacheinander alle Positionen. Das funktioniert, weil die Methode `children()` alle Kindelemente von Positionen – also dem Elternelement der einzelnen Positionen – findet. Bei jedem Schleifendurchlauf wird eine Position in der Variablen `$position` zur Verfügung gestellt. Auf die Elemente dieser gerade aktuellen Position wird dann ganz normal über den Operator `->` und den Elementnamen zugegriffen – `$position->Artikel`.

Selbstverständlich können Sie mit den Daten aus der XML-Datei auch rechnen und arbeiten. Wir berechnen hier zum Beispiel für jede Position aus der Menge und dem Preis die Summe für die jeweilige Rechnungsposition.

```
$summepos = floatval($position->Preis) * $position->Anzahl;
```

Zudem schreiben wir die gesamte Rechnungssumme mit

```
$gesamtpreis += $summepos;
```

und geben diese später ein wenig formatiert ebenfalls mit aus.

Die Ausgabe dieses Teils erfolgt in einer Tabelle. Lassen Sie sich durch die vielen HTML-Tags, die dem Ausgabestring hinzugefügt werden, an dieser Stelle nicht irritieren.

Ganz zum Schluss geben wir mit der Anweisung

```
echo $ausgabe;
```

den zuvor erstellten Ausgabestring nur noch aus. Damit ist das Skript abgeschlossen.

Für einen ersten Eindruck, wie die Zusammenarbeit zwischen PHP-Skripten und XML-Dokumenten erfolgen kann, soll das an dieser Stelle ausreichen. Andere PHP-Erweiterungen bieten noch andere und umfangreichere Möglichkeiten zum Arbeiten mit XML-Dokumenten. Ein Blick in das PHP-Handbuch<sup>26</sup> liefert Ihnen bei Bedarf umfangreiche Informationen.

Fassen wir das Wichtigste noch einmal zusammen.

## Zusammenfassung

XML steht für eXtensible Markup Language. Es wurde 1998 durch das W3C als Standard zur Dokumentenauszeichnung eingeführt. XML dient dazu, Daten in strukturierter Form als Text zu speichern.

XML ist ein offener Standard und lizenzfrei.

Ein XML-Dokument ist hierarchisch aufgebaut. Es besteht aus dem Prolog und den eigentlichen Daten.

PHP stellt bereits in der Standardinstallation mehrere Erweiterungen zur Verfügung, um mit XML-Dokumenten zu arbeiten.

## Aufgaben zur Selbstüberprüfung

- 5.1 Nennen Sie drei Attribute der XML-Deklaration. Unterscheiden Sie diese auch in obligatorische und optionale Attribute.
- 5.2 Nennen Sie eine wichtige Regel für die Angabe der XML-Deklaration.
- 5.3 Was ist das Root-Element einer XML-Datei?
- 5.4 Wie lautet die Standard-Zeichensatzcodierung von XML-Dokumenten?
- 5.5 Nennen Sie den Namen einer PHP-Erweiterung zum Arbeiten mit XML-Dokumenten.

26. <http://php.net/manual/de/index.php>

## Schlussbetrachtung

In diesem Studienheft haben Sie sich die Grundlagen erarbeitet, um erfolgreich einen Einstieg in die Entwicklung von Webanwendungen zu unternehmen. Sie wissen nun, was Webanwendungen sind, kennen die Vorteile und wissen auch, welche Herausforderungen auf Sie warten.

Sie haben sich einen Überblick über Techniken der serverseitigen Programmierung verschafft und mit PHP die am weitesten verbreitete Skriptsprache für serverseitige Programmierung im Internet kennengelernt. An einigen Beispielen konnten Sie die Funktionsweise von PHP bereits nachvollziehen.

Nachdem Sie sich einige Grundlagen von MySQL erarbeitet hatten, haben Sie PHP und MySQL kurz im Zusammenspiel erlebt.

Wenn Sie sich nach dem Bearbeiten dieses Hefts entscheiden sollten, ein wenig mit PHP zu spielen und erste Anwendungen zu entwickeln, dann kennen Sie jetzt bereits die wichtigsten Editoren, integrierten Entwicklungsumgebungen und Frameworks.

Nicht immer ist die Datenquelle einer Anwendung eine Datenbank. Mit XML-Dokumenten haben Sie eine alternative Möglichkeit kennengelernt. Sie wissen jetzt, wie ein XML-Dokument aufgebaut ist und wie einfach Sie mit PHP darauf zugreifen können.

Auf dem Gelernten können Sie sehr gut aufbauen, sollten Sie vorhaben, sich zukünftig intensiver mit PHP zu beschäftigen und selbst erste eigene Webanwendungen zu programmieren.

Torsten Schreiber



## A. Lösungen der Aufgaben zur Selbstüberprüfung

Hier finden Sie die Lösungen zu den Aufgaben zur Selbstüberprüfung in den einzelnen Kapiteln. Bei offenen Aufgaben mit freien Formulierungen kommt es nicht auf eine wörtliche Übereinstimmung an, sondern auf den Inhalt. Entsprechen Ihre Ergebnisse nicht den Lösungen, wiederholen Sie bitte das entsprechende Kapitel und bearbeiten Sie die zugehörigen Aufgaben zur Selbstüberprüfung nach einer Pause erneut.

### Kapitel 1

- 1.1 JavaScript ist die wichtigste Skriptsprache im Zusammenhang mit clientseitiger Programmierung. Den Namen erhielt JavaScript, da es syntaktische Ähnlichkeit zur Programmiersprache Java hat.
- 1.2 Die drei Schichten sind:
  - 1. Präsentationsschicht
  - 2. Applikationslogik
  - 3. Datenbankschicht
- 1.3 CGI belastet den Server sehr, da für jeden CGI-Aufruf ein neuer Prozess mit eigener Hauptspeicherreservierung aufgerufen wird.

### Kapitel 2

- 2.1 Sie öffnen einen PHP-Block mit `<?php` und schließen ihn wieder mit `?>`.
- 2.2 Der PHP-Interpreter führt eine Syntaxanalyse durch. Dabei zerlegt er den Quelltext der Datei und bringt ihn in eine für die Ausführung geeignete Form (parsen). Anschließend wird der PHP-Code ausgeführt.
- 2.3 Der Webserver erkennt nicht am Inhalt der Datei, sondern an der Namenserverweiterung `.php`, dass die Datei durch den PHP-Interpreter verarbeitet werden muss.
- 2.4 Ein Kommando lautet `echo`. Eine Alternative wäre `print`.

### Kapitel 3

- 3.1 MySQLi ist die neuere Erweiterung von PHP zum Zugriff auf MySQL-Datenbanken, die neben der prozeduralen Programmierung vor allem auch die objektorientierte Programmierung erlaubt. Das `i` steht für improved und heißt verbessert.
- 3.2 RDBMS steht für Relationales Datenbankmanagementsystem.
- 3.3 MySQL und MySQLi stehen exklusiv für Verbindungen zu MySQL zur Verfügung, wogegen mit PDO verschiedene Datenbanksysteme verwendet werden können. Zudem ist PDO rein objektorientiert.

### Kapitel 4

- 4.1 Editoren zum Beispiel: Notepad++ und Sublime Text  
IDEs zum Beispiel: Aptana Studio und PhpStorm

- 4.2 Möglichkeit eines Editors, bestimmte Begriffe und Zeichenkombinationen abhängig von ihrer Bedeutung in unterschiedlichen Stilen, Schriftarten und Farben darzustellen.
- 4.3 Ein Debugger ist eine Software, die zur Laufzeit eine Anwendung an Haltepunkten unterbrechen und dann schrittweise bestimmte Passagen durchlaufen kann. Währenddessen werden die Werte und Zustände von Variablen und Objekten angezeigt.

### Kapitel 5

- 5.1 Obligatorisch: `version`  
Optional: `encoding` und `standalone`
- 5.2 Die XML-Deklaration muss direkt zu Beginn der Datei stehen. Es dürfen weder Leerzeilen noch Leerzeichen davor stehen.
- 5.3 Es befindet sich ganz oben in der Hierarchie. Es gibt nur ein Root-Element in einer XML-Datei. Alle weiteren Elemente müssen sich innerhalb dieses einen Elements befinden.
- 5.4 UTF-8
- 5.5 Zum Beispiel die Erweiterung SimpleXML, mit der in diesem Studienheft gearbeitet wurde.

## B. Glossar

Active Server Pages (ASP)	Eine Technologie, um Code-Elemente in HTML einzubetten, auf dem Server auszuführen und so dynamische Webseiten zu erzeugen. Skriptsprachen: JScript oder VB-Skript.
Ajax	Steht für Asynchronous JavaScript And XML. Ajax ist keine eigenständige Technologie, sondern vereint mehrere in der Webprogrammierung bekannte Technologien unter einem gemeinsamen Namen und in einem gemeinsamen Ansatz. Besonderes Merkmal ist die asynchrone Datenübertragung.
Apache	Der Apache Webserver ist freie Software. Er ist der weltweit meistgenutzte Webserver im Internet.
Applikationsserver	Ein spezieller Server, auf dem (Web-)Anwendungen ausgeführt werden.
ASP.NET	Active Server Pages .NET ist eine auf dem .NET-Framework von Microsoft basierende Technologie zum Entwickeln dynamischer Webseiten. Mit ASP.NET können Webanwendungen in allen Programmiersprachen entwickelt werden, die durch das .NET-Framework unterstützt werden – beispielsweise Visual Basic oder C#.
Asynchrone Datenübertragung	Bedeutet, dass der Benutzer weiterarbeiten kann, während im Hintergrund Daten ausgetauscht werden. Das wird erreicht, weil nicht die gesamte Seite neu geladen werden muss, sondern nur die relevanten Teile einer Seite ausgetauscht werden.
Asynchronous JavaScript And XML	→ Ajax
CGI	Abkürzung für Common Gateway Interface. Der älteste und bis heute weitverbreitete Ansatz für serverseitige Dynamik. Ermöglicht das Ausführen von Programmen auf dem Webserver. Spezifiziert die erforderlichen Schnittstellen für die Eingabe und Weiterleitung von Informationen vom Web-Client zum Programm und für die Rückgabe zum Client.
Clientseitige Programmierung	Skripte und Programme, die auf dem Webclient ausgeführt werden.
Common Gateway Interface	→ CGI
DBMS	Abkürzung für Datenbankmanagementsystem

FastCGI	Eine Erweiterung zu CGI, die den Performancenachteil von CGI ausgleichen soll.
Hidden-Field	Verstecktes Feld in einem Formular.
HTTP	Steht für Hypertext Transport Protokoll. HTTP ist ein zustandsloses Protokoll zur Datenübertragung im World Wide Web.
Hypertext Transport Protokoll	→ http
IIS	steht für Internet Information Services (vormals Internet Information Server). Der IIS ist ein häufig eingesetzter Webserver von Microsoft.
Index	Über Indizes lässt sich die Suche nach Daten in der Datenbank beschleunigen.
Java Server Pages (JSP)	Auf Java basierende Technologie, um Code-Elemente in HTML einzubetten, auf dem Server auszuführen und so dynamische Webseiten zu erzeugen.
JavaScript	Wichtigste Skriptsprache für clientseitige Programmierung.
localhost	Steht für das aktuell verwendete System oder dessen IP-Adresse
MySQL	MySQL ist eines der weltweit führenden relationalen Datenbankmanagementsysteme (RDBMS) im Zusammenhang mit dynamischen Internetauftritten.
MySQLi	Etwas modernere PHP-Erweiterung für die Zusammenarbeit mit MySQL. Das i steht für improved (dt. verbessert).
Netzprotokoll	→ Netzwerkprotokoll
Netzwerkprotokoll	Ein Protokoll regelt den Austausch von Daten zwischen Computern und/oder Diensten.
PHP	PHP (PHP Hypertext Preprocessor) ist eine extra für die Web-Programmierung entwickelte und nahezu exklusiv dort eingesetzte Skriptsprache.
phpinfo()	PHP-Funktion, um Details zur aktuellen PHP-Installation abzurufen.
PHP-Interpreter	Verarbeitet den PHP-Code.
phpMyAdmin	Administrationsprogramm mit grafischer Oberfläche für MySQL.

PHP-Parser	Der Parser wandelt den eingebetteten PHP-Code in eine für den Interpreter verarbeitbare Form um.
Prozedurale Programmierung	Bei der prozeduralen Programmierung wird eine Folge von Anweisungen in einem Programm ausgeführt. Zusammengehörende Anweisungen werden dabei in kleineren Einheiten (Prozeduren) zusammengefasst.
Objektorientierte Programmierung (OOP)	Eine Technik oder Programmiersprache, die Objekte, Klassen und Vererbung unterstützt.
Relation	Eine Tabelle in einem relationalen Datenbankmanagementsystem.
Rich Internet Applications	Besondere Webanwendungen, die in ihren Interaktionsmöglichkeiten und im gesamten Erscheinungsbild Desktopanwendungen entsprechen.
SCGI	Ähnlich FastCGI
Server Side Includes (SSI)	Bei SSI handelt es sich um Skriptbefehle, die in HTML eingebettet und beim Aufruf der Seite durch ein Webserver-Modul ausgeführt werden.
Serverseitige Programmierung	Unter serverseitiger Programmierung versteht man Programme und Skripte, die auf einem Web- oder Applikationsserver ausgeführt werden.
Servlets	Servlets sind Java-Klassen, deren Instanzen auf Webservern dynamisch Anfragen von Webclients beantworten können.
Session	Als Session (dt. Sitzung) wird eine stehende Verbindung zwischen einem Client und einem Server bezeichnet. Bei Webanwendungen kann eine Session wegen der Verwendung von HTTP erst auf Anwendungsebene zustande kommen.
Synchrone Datenübertragung	Benutzeraktivität und Verarbeitung auf dem Server wechseln sich ab.
Variable	Eine Variable ist ein Datenobjekt, das seinen Wert verändern kann.
Webanwendung	Eine Webanwendung ist eine Softwareanwendung, die auf einem Web- oder Applikationsserver im Internet bzw. Intranet ausgeführt wird, in der Regel über eine Datenbankbindung verfügt, Webseiten dynamisch erzeugt und dem Benutzer die Interaktion mit der Anwendung und/oder den Anwendungsdaten ermöglicht.
Webapplikation	→ Webanwendung

Webclient	Ein typischer Webclient ist der Webbrowser. Mit ihm werden Inhalte vom Webserver über das Internet angefordert.
Webserver	Eine Software, die Inhalte bereitstellt, die von Webclients angefordert werden können.
XHTML	XHTML (Extensible HyperText Markup Language) ist eine Neuformulierung von HTML auf Basis von XML.
XML	Steht für Extensible Markup Language. XML ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten als Textdaten. XML erlaubt den plattformunabhängigen Datenaustausch zwischen Systemen.
XMLHttpRequest	Das ist eine Programmierschnittstelle, die es erlaubt, mit JavaScript HTTP-Anforderungen (Requests) an einen Server zu senden, ohne dass die Seite dabei neu geladen werden muss.
Zustandslos	Unter Zustandslosigkeit versteht man im Zusammenhang mit Netzwerkprotokollen die Eigenschaft, Anfragen komplett unabhängig voneinander zu behandeln. Das bedeutet, dass mehrere Anfragen desselben Clients oder Auftraggebers nicht in Bezug zueinander stehen.

## C. Literaturverzeichnis

Wenz, C. & Hauser, T. (2016). *PHP 7 und MySQL: Das umfassende Handbuch*. 2. Auflage. Bonn: Rheinwerk Computing.

Theis, T. (2016). *Einstieg in PHP 7 und MySQL 5.6. Ideal für Programmieranfänger geeignet*. 11. Auflage. Bonn: Rheinwerk Computing.

### Internet-Links

PHP Handbuch

<http://php.net/manual/de/index.php>

Deutschsprachiges PHP-Forum

<http://phpforum.de/forum/>

w3techs – Statistiken zur Verwendung von Programmiersprachen

[http://w3techs.com/technologies/overview/programming\\_language/all](http://w3techs.com/technologies/overview/programming_language/all)

Publikation zur Sicherheit von Webanwendungen

<https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/WebSec/WebSec.pdf>

MySQL Beispieldatenbanken

<http://dev.mysql.com/doc/index-other.html>

## D. Abbildungsverzeichnis

Abb. 1.1	Infrastruktur für einfache Webanwendungen .....	4
Abb. 1.2	3-Schicht-Architektur für umfangreiche Webanwendungen .....	4
Abb. 1.3	Abruf statischer Webseiten .....	5
Abb. 1.4	Dynamisch erzeugte Webseiten .....	6
Abb. 1.5	Wikipedia als Beispiel für eine datenbankgestützte Webanwendung ....	7
Abb. 1.6	Suche mit Ajax-Technologie .....	10
Abb. 2.1	Funktionsweise von PHP .....	18
Abb. 2.2	„Hallo Welt!“ im Browser .....	20
Abb. 2.3	Ausgabe als Überschrift .....	21
Abb. 2.4	Die Ausgabe der Berechnungen im Browser .....	22
Abb. 2.5	Absatzweise Ausgaben .....	23
Abb. 2.6	Die Ausgabe der Potenz (am unteren Rand) .....	25
Abb. 2.7	Die Ausgabe von phpinfo() im Browser .....	26
Abb. 2.8	Von phpinfo() generierter Quelltext.....	26
Abb. 2.9	Skript nach der Ausführung .....	28
Abb. 2.10	PHP-Dateien im Dateisystem.....	30
Abb. 2.11	Ein einfaches Formular .....	31
Abb. 2.12	Die Auswertung des Formulars.....	33
Abb. 3.1	Struktur und Beziehungen von Tabellen einer Datenbank in phpMyAdmin.....	36
Abb. 3.2	Das Ergebnis der Zusammenarbeit: Daten von über 200 Ländern .....	41
Abb. 3.3	Formatierte Ausgabe des Ergebnisses von zuvor .....	42
Abb. 4.1	PHP-Datei im Windows Editor .....	45
Abb. 4.2	Geöffnete PHP-Datei in Sublime Text .....	45
Abb. 4.3	Syntax-Highlighting im Notepad++ .....	46
Abb. 4.4	Aptana Studio 3 mit geöffnetem Projekt .....	47
Abb. 4.5	PhpStorm mit Projektverwaltung, Editor und Datenbankansicht .....	49
Abb. 4.6	Verfügbare Plugins für PhpStorm .....	49
Abb. 4.7	Symfony Plugin zur Framework-Integration direkt aus PhpStorm installieren .....	50
Abb. 4.8	Debugging in Aptana Studio 3 .....	51
Abb. 4.9	Die 15 „besten“ kostenlosen PHP-Frameworks im Jahr 2015 .....	54
Abb. 5.1	Ausgabe im Browser .....	61
Abb. 5.2	Zugriff auf die Elemente .....	64



## E. Quellcodeverzeichnis

Code 2.1	Hallo Welt .....	19
Code 2.2	Hallo Welt 2 .....	21
Code 2.3	Rechnen .....	22
Code 2.4	Rechnen mit korrigierter Ausgabe .....	23
Code 2.5	Verwendung der Funktion pow() .....	24
Code 2.6	phpinfo() .....	25
Code 2.7	Verschiedener PHP-Code an mehreren Stellen .....	28
Code 2.8	Das Formular .....	31
Code 2.9	Die Auswertung des Formulars .....	33
Code 3.1	Beispiel für die vier Schritte bei der Zusammenarbeit zwischen PHP und MySQL .....	39
Code 5.1	Beispiel einer einfachen XML-Datei (rechnung.xml) .....	58
Code 5.2	Eine XML-Datei mit PHP lesen .....	63

## F. Sachwortverzeichnis

<b>Numerics</b>		
3-Schicht-Architektur .....	4	
<b>A</b>		
Active Server Pages .....	11	
Ajax .....	9	
Ajax-Anwendungen .....	8	
Applikationsserver .....	4	
Array		
assoziativ .....	33	
ASP.NET .....	11	
asynchrone Datenübertragung .....	9	
Auto-Completion .....	46	
Autovervollständigen .....	46	
<b>C</b>		
CGI .....	11	
Clientseitige Programmierung .....	8	
Code-Hinting .....	46	
Community Edition .....	35	
Cookies .....	13	
Cross-Site-Scripting .....	14	
<b>D</b>		
Datenbank .....	10	
Datenbankmanagementsysteme .....	4	
Debugger .....	48	
Document Object Model .....	9	
Dokumenttypdefinition .....	59	
Dokumenttypdeklaration .....	59	
DTD .....	59	
<b>E</b>		
End-Tag .....	60	
eXtensible Markup Language .....	57	
<b>F</b>		
FastCGI .....	11	
<b>H</b>		
HTML .....	5	
Hypertext Markup Language .....	5	
<b>I</b>		
IDE .....	47	
Injection .....	13	
Integrated Development		
Environment .....	47	
integrierte Entwicklungsumgebung ..	47	
Interpreter .....	29, 30	
<b>J</b>		
Java Server Pages .....	11	
JavaScript .....	8	
<b>K</b>		
kompilierte Programme .....	10	
<b>L</b>		
Leeres Element .....	60	
<b>M</b>		
MariaDB .....	35	
MySQL .....	37	
MySQLi .....	37	
<b>N</b>		
Netzwerkprotokoll .....	3	
Notepad++ .....	45	
<b>P</b>		
parsen .....	18	
Parser .....	30	
PDO .....	37	
Personal Home Page/Forms		
Interpreter .....	16	
PHP .....	16	
PHP Data Objects .....	37	
PHP Hypertext Preprocessor .....	16	
PHP/FI .....	16	
Plugins .....	49	
prepared statements .....	13	
Projekt .....	47	
Prolog .....	58	
Protokollfamilie .....	3	

**R**

Request .....	5
Response .....	5
RIA .....	7
Rich Internet Applications .....	7
Root-Element .....	59

**S**

SCGI .....	11
serverseitige Webprogrammierung ...	10
Servlets .....	11
Sessions .....	13
SGML .....	57
Sicherheit .....	13
Skriptsprachen .....	10
SQL .....	36
SQL-Injection .....	13
Stammelement .....	59
Standard Generalized Markup Language .....	57
Start-Tag .....	60
Structured Query Language .....	36
Sublime Text .....	45
synchrone Datenübertragung .....	9
Syntaxanalyse .....	18
Syntaxhervorhebung .....	46
Syntax-Highlighting .....	46

**T**

TCP/IP .....	3
--------------	---

**V**

Variable .....	22
vorbereitete Anweisungen .....	13

**W**

Webanwendung .....	3
Webapplikation .....	3
Webbrowser .....	3
Webclient .....	3, 5
Webprogrammierung .....	8
Webserver .....	5
Wurzelement .....	59

**X**

XML .....	57
XML-Deklaration .....	58
XMLHttpRequest .....	10
XSS Cross-Site-Scripting .....	14

**Z**

zustandsloses Netzwerkprotokoll .....	13
---------------------------------------	----



## G. Einsendeaufgabe

### Serverseitige Programmierung

Code:  
**MMDE12C-XX1-N01**

Name:	Vorname:
Postleitzahl und Ort:	Straße:
Studien- bzw. Vertrags-Nr.:	Lehrgangs-Nr.:

Fernlehrer/in:
Datum:
Note:
Unterschrift Fernlehrer/in:

Bitte reichen Sie Ihre Lösungen über die Online-Lernplattform ein oder schicken Sie uns diese per Post. Geben Sie bitte immer den Code zum Studienheft an (siehe oben rechts).

1. Beschreiben Sie kurz den Ablauf einer Webanwendung in einer Three-Tier-Architecture?

**8 Pkt.**

2. Erläutern Sie kurz die Begriffe Request und Response.

**4 Pkt.**

3. Welche der folgenden Aussage(n) in Bezug auf Ajax ist/sind richtig?

- a) Wesentliches Merkmal von Ajax ist der synchrone Datenaustausch.
- b) Der Datenaustausch erfolgt bei Ajax über ein Objekt der Klasse XMLHttpRequest.
- c) Einzelne Bereiche einer Seite werden mit PHP und dem DOM aktualisiert.

**6 Pkt.**

4. Welche der folgenden Aussage(n) in Bezug auf CGI ist/sind richtig?

- a) CGI ist eine Protokollvereinbarung.
- b) CGI ist die älteste Programmiersprache für serverseitige Programmierung.
- c) CGI ist der Vorgänger von PHP.
- d) Für jeden CGI-Aufruf wird auf dem Webserver ein neuer Prozess gestartet.
- e) CGI-Programme können mit verschiedenen Programmiersprachen entwickelt werden.

**10 Pkt.**

5. Erläutern Sie kurz, warum die Zustandslosigkeit von http eine Herausforderung bei der Entwicklung von Webanwendungen ist und wie Sie dieser Herausforderung begegnen können.

**8 Pkt.**

6. Was versteht man unter XSS?

**8 Pkt.**

7. Erläutern Sie mit eigenen Worten die grundsätzliche Funktionsweise von PHP.

**10 Pkt.**

8. Nehmen wir an, Sie haben Probleme beim Ausführen einer PHP-Datei. Das darin enthaltene Skript wird einfach nicht ausgeführt. Beim Blick in die Adresszeile des Browsers sehen Sie, dass der Eintrag dort mit `file://` beginnt. Erläutern Sie, warum das PHP-Skript nicht ausgeführt werden konnte.

**5 Pkt.**

9. Erläutern Sie kurz mit eigenen Worten, wie in PHP auf gesendete Formulardaten zugegriffen wird.

**5 Pkt.**

10. Sie möchten PHP 7 zusammen mit einer MySQL-Datenbank benutzen. Welche PHP-Erweiterungen können Sie dabei einsetzen?

**6 Pkt.**

11. Beschreiben Sie mit eigenen Worten den grundsätzlichen Ablauf einer Zusammenarbeit zwischen PHP und MySQL.

**12 Pkt.**

12. Nennen Sie die beiden großen Vorteile, die für die grundsätzliche Verwendung einer IDE sprechen.

**6 Pkt.**

13. In das folgende kleine PHP-Skript haben sich drei Fehler eingeschlichen. Welche?

```
<php
    phpinfo[ ],
?>
```

**6 Pkt.**

14. Nennen Sie mindestens drei grundsätzliche Vorteile beim Einsatz eines PHP-Frameworks.

**6 Pkt.**

**Gesamt 100 Pkt.**