Einführung in die maschinennahe, imperative, funktionale, relationale und objektorientierte Programmierung

-

EMIFROP 0.26

Markus Alpers
B.Sc. und Ausbilder f. Industriekaufleute

1. März 2016

Inhaltsverzeichnis

Hinweis bezüglich diskriminierender Formulierungen

In diesem Text wurde darauf geachtet Formulierungen zu vermeiden, die diskriminierend verstanden werden können. Im Sinne der Lesbarkeit wurden dabei Formulierungen wie „Informatiker und Informatikerinnen“ durch „InformatikerInnen“ (mit großem i) ersetzt. An anderen Stellen habe ich Formen wie eine/einer durch eineR zusammengefasst. Hier berufe ich mich auf den Artikel „Sprache und Ungleichheit“ der Bundeszentrale für politische Bildung, kurz BpB, vom 16. April 2014, insbesondere auf den Absatz „Zum Umgang mit diskriminierender Sprache“, online abrufbar unter:

`http://www.bpb.de/apuz/130411/sprache-und-ungleichheit?p=all`

Sollten Sie dennoch Formulierungen entdecken, die diesem Anspruch nicht entsprechen, möchte ich Sie bitten, mir eine entsprechende Nachricht zu senden, denn es ist mir wichtig, Ihnen mit diesem Buch eine wertvolle Unterstützung beim Start in die faszinierende Welt der Informatik zu bieten. Das sollte nicht durch verletzte Gefühle in Folge missverständlicher Formulierungen torpediert werden.

Sie erreichen mich unter `markus.alpers@haw-hamburg.de`.

Hinweis zur Lizenz

zenz"´. Wenn Sie einen Teil finden, in dem diese „Hinweise zur Lizenz"´ nicht zu finden ist, dann dürfen Sie für den persönlichen Gebrauch dennoch Kopien davon anfertigen und Sie dürfen diese Kopien außerhalb von kommerziellen Projekten frei verwenden.

Hinweis zur Verwendbarkeit in wissenschaftlichen Arbeiten

Bitte beachten Sie dabei aber, dass die Verwendung dieses Textes im Rahmen wissenschaftlicher Publikationen zurzeit aus anderen Gründen problematisch ist: Wie viele andere Quellen, die frei im Internet verfügbar sind, wurde auch dieser Text bislang nicht durch einen nachweislich entsprechend qualifizierten Lektor verifiziert. Damit genügen Zitate aus diesem Band streng genommen noch nicht den Ansprüchen wissenschaftlicher Arbeiten.

Bitte beachten Sie außerdem, dass dieses Buch eine Konvention nutzt, die in wissenschaftlichen Arbeiten verpönt ist: Wenn in einer wissenschaftlichen Arbeit ein Begriff hervorgehoben wird, dann wird dazu kursive Schrift verwendet. In diesem Buch verwende ich dagegen Fettdruck, da es vielen Menschen schwer fällt, einen kursiv gedruckten Begriff schnell zu finden und ich mir wünsche, dass Sie es möglichst effizient auch als Nachschlagewerk nutzen können.

Teil I

Einfache Einführung in die imperative Programmierung

Kapitel 1

Typische Irrtümer darüber, was Programmieren ist.

Kapitel 2

Von der Nachrichtentechnik zur Programmierung

Kapitel 3

Vorbereitung fürs Programmieren

Kapitel 4

Ausgewählte Programmiersprachen

Kapitel 5

Grundlagen der Entwicklung verteilter Anwendungen

Verteilte Anwendungen sind Programme, die aus einer Vielzahl von einzelnen und unabhängigen Programmen bestehen, die jeweils auf unterschiedlichen Rechnern als Server bzw. Clients in einem Netzwerk Teilaufgaben erfüllen. Auch wenn Sie unabhängig voneinander agieren, sind Sie doch als Ganzes eine gemeinsame Funktionalität bzw. ein gemeinsames Programm. Erst wenn dieses letzte Kriterium gilt, wird das Ganze als eine verteilte Anwendung bezeichnet.

Wenn Ihnen das zu abstrakt klingt, dann stellen Sie sich ein Team von verschiedenen MitarbeiterInnen vor, die an verschiedenen Standorten für ein Unternehmen tätig sind. Die verschiedenen Server und Clients entsprechen den MitarbeiterInnen. Wenn Sie mit den Begriffen Server und Client Schwierigkeiten haben, dann lesen Sie bitte die entsprechenden Abschnitte der vorhergehenden Kapitel.

Die einfachste Möglichkeit, um eine verteilte Anwendung zu entwickeln besteht in einer Kombination, bei der HTML für die Struktur der Anwendung verwendet wird. Zusätzlich benötigen wir dann noch eine Sprache um die Funktionalität der Elemente dieser Anwendung zu programmieren und abschließend eine Sprache, um die langfristige Speicherung von Nutzerdaten sicher zu stellen. Das können Sie als Erstsemester mit entsprechendem Arbeitsaufwand gut schaffen.

„Echte“ verteilte Anwendungen werden dagegen erst im Masterbereich eines Informatikstudiums grundlegend behandelt, weil Sie für den Einstieg in diesen Bereich eine Vielzahl an Kursen absolviert haben müssen, die Teil eines Bachelorstudiums der Informatik und verwandter Studiengänge ist. Hier seien die wichtigsten dieser Veranstaltungen und Themenbereiche ge-

nannt:

- Diskrete Mathematik und Graphentheorie
- Grundlagen der technischen Informatik
- Nachrichten- und Kommunikationstechnik
- Netzwerke und Internetsicherheit
- Imperative Programmierung
- Objektorientierte Softwareentwicklung
- (Relationale) Datenbanken
- Mobile Apps und Responsive Design
- Medienrecht

Wie eingangs beschrieben können Sie einfache verteilte Anwendungen wie beispielsweise eine Webanwendung auch ohne fundierte Kenntnisse dieser Bereiche entwickeln. Leider kann es Ihnen passieren, dass Sie nach dem Studienabschluss in einem Unternehmen tätig werden, wo Sie es mit vermeintlichen Fachkräften zu tun haben, deren Kenntnisse der oben genannten Bereiche zumindest teilweise mangelhaft sind. Diese wissen dann z.B., dass es so etwas wie „das Internet“ gibt, wie sie eine Webanwendung programmieren können und ähnliches, aber ihnen fehlt das fundamentale Verständnis für die zugrunde liegenden Technologien. Das führt dann langfristig zu massiven Schwierigkeiten, denn sobald neue Technologien auf dem Markt erscheinen (HTML 5 ist da ein sehr gutes Beispiel) sind diese „Fachkräfte“ nicht im Stande die neuen Technologien sinnvoll in bestehende Projekte zu integrieren und was recht bald dazu führt, dass Sie einen neuen Arbeitgeber benötigen.

Bevor Sie hier weiterlesen: Haben Sie die einleitenden Kapitel zumindest überflogen und den Abschnitt zur Vorbereitung für die Entwicklung von verteilten Anwendungen durchgearbeitet? Wenn nicht, dann tun Sie das bitte, bevor Sie hier weitermachen. Im laufenden Kapitel wird zum einen vorausgesetzt, dass Sie die Inhalte der einleitenden Kapitel kennen und insbesondere die nötigen Installationen abgeschlossen haben. Unklare Fachbegriffe können sie anhand der Anhänge zu jedem Kapitel leicht nachschlagen.

Wenn Sie ein anderes Paket als EasyPHP nutzen, dann werden einige Ausgaben des Rechners bei Ihnen anders aussehen als hier geschildert. Eine zusätzliche Hilfe für diese Fälle kann leider im Rahmen dieses Kurses nicht

geboten werden.

Einführende Programmierveranstaltungen bereiten Sie in aller Regel darauf vor, Programme für einen Computer zu entwickeln. Das ist in dieser Veranstaltung anders: Hier lernen Sie, Programme zu entwickeln, die über das World Wide Web auf allen möglichen Endgeräten laufen können. Mit Endgeräten ist alles gemeint, was im Kern ein Computer ist, aber nicht unbedingt als solcher zu erkennen. Ein Beispiel sind Smartphones.

5.1 Internet, WWW und HTML

In einer Veranstaltung zur **Nachrichten- oder Kommunikationstechnik** werden Sie lernen, dass das, was allgemein als „das Internet“ bezeichnet wird in Wahrheit etwas ist, das mit dem Begriff „World Wide Web“ bezeichnet wird. Der Begriff **Internet** bezeichnet dagegen Verbindungen zwischen zwei Netzwerken auf globaler Ebene. Ein Beispiel hierfür wären die transatlantischen Kabel, die die Verbindung zwischen den Kommunikations- und Datennetzen in Europa und Amerika sicherstellen.

Somit ist das Internet also die Voraussetzung für das World Wide Web: Ohne diese Leitungen und Funkstrecken (z.B. über Satellit) könnten wir keine Daten in ferne Länder übertragen oder von dort erhalten. Und Psys Gangnam Style wäre nie so berühmt geworden. Das WWW ist damit die wahrscheinlich umfangreichste verteilte Anwendung, die es weltweit gibt.

Doch was ist nun das **World Wide Web** (kurz **WWW**)? Hier handelt es sich um Dateien, die auf Computern rund um die Welt gespeichert sind. Diese Daten sind in bestimmten Formaten verfasst und dienen dazu, mit einem entsprechenden Programm, Texte und multimediale Inhalte anzuzeigen. Das Format in dem diese Dateien verfasst sind ist eine von mehreren Programmiersprachen, die in aller Regel auf die beiden Buchstaben ML endet. Dieses ML steht für Markup Language. Eine **Markup Language** wird von den meisten Softwareentwicklern nicht als „echte“ Programmiersprache anerkannt. Sie können mit einer ML nämlich lediglich Elemente oder sogenannte Container definieren, die Texte oder Verweise auf beliebige Daten beinhalten. Interaktive Programme (also Anwendungen, die auf die Eingaben von Nutzern so reagieren, dass sich ihre Inhalte ändern) sind in HTML wie in beliebigen anderen Markup Languages nicht realisierbar.

An dieser Stelle müssen wir uns drei Dinge klarmachen:

1. Die Inhalte (also Texte, Bilder, Hintergrundmusik, Videos usw. usf.)

sind NICHT Teil einer Markup Language. Vielmehr gibt es innerhalb einer Markup Language verschiedene Möglichkeiten, um auf den Speicherort zu verweisen, an dem diese Inhalte sich befinden. Das kann auch durch eine weitere Programmiersprache passieren. Allerdings ist es meist möglich, Texte direkt innerhalb einer Markup Language einzutragen.

2. Markup Languages sind statisch, weil Sie ausschließlich dazu dienen, Strukturen zu definieren. Alles dynamische wird in einer zweiten Programmiersprache programmiert.

3. In einer Markup Language benutzen wir sogenannte Tags (sprich Täg), um zu definieren, was für eine Art von Element wir definieren wollen. In HTML5 benutzen wird beispielsweise das Tag `<main>`, um einen Bereich zu definieren, der zentral für unsere Webanwendung ist. Wie dieser Bereich dann später angezeigt wird, ist für die Programmierung in HTML vollkommen irrelevant.

Allerdings ist die Behauptung, dass eine Markup Language keine Programmiersprache ist schlichtweg falsch. Richtig ist dagegen, dass es weder eine imperative noch eine deklarative Programmiersprache ist. Richtig ist aber auch und insbesondere, dass Markup Languages außerordentlich wichtig sind, wenn wir es mit großen Softwareprojekten zu tun bekommen. In der objektorientierten Softwareentwicklung wird beispielsweise die **UML** (kurz für Unified Markup Language) genutzt, um übersichtlich darzustellen, aus welchen Komponenten ein Softwareprojekt besteht. Die UML ist aber noch zu wesentlich mehr zu gebrauchen, doch das würde jetzt vom Thema ablenken.

Markup Languages sind also ein wichtiges Werkzeug, wann immer wir ein umfangreiches Softwareprojekt realisieren wollen. Sie helfen uns, Fehler zu vermeiden und klare Strukturen im Projekt zu schaffen, wodurch die Teamarbeit deutlich erleichtert wird.

Programme, die Dateien in einer Markup Language, nämlich HTML anzeigen können kennen Sie, denn Sie nutzen Sie täglich: Es sind Webbrowser. Aber damit die Dateien von anderen Computern auf Ihrem Endgerät angezeigt werden können ist es noch nötig, dass sie von dort auf Ihr Endgerät übertragen werden. Wenn Sie also bislang dachten, dass es einen echten Unterschied zwischen einem Download und dem Besuch einer Webanwendung (bzw. einer Webpage) gibt, dann wissen Sie es jetzt besser: Alles, was in Ihrem Webbrowser angezeigt wird liegt mindestens so lange im Speicher des Endgerätes, wie es angezeigt wird. Denn für einen Computer macht es im Grunde keinen Unterschied, ob Daten „nur“ im (RAM-)Speicher oder

als sogenannte Datei auf einem Speicher wie einer Festplatte vorliegen.

5.1.1 HTTP und HTTPS

Damit Daten auf Ihr Endgerät übertragen werden gibt es verschiedene Verfahren, von denen Sie als Nutzer eines Gerätes nichts sehen. Diese Verfahren werden in Form sogenannter Protokolle vereinbart. Wie die einzelnen Protokolle aussehen, das ist Teil der Veranstaltung Netzwerke und Internetsicherheit. Für diesen Kurs genügt es, dass Sie wissen, was ein **Protokoll** ist. Eines dieser Protokolle übersehen Sie praktisch jedes Mal, wenn Sie eine Webanwendung aufrufen: Es handelt sich um das **HTTP**, das **HyperText Transfer Protocol**. Das ist ein Protokoll, das einzig dazu entwickelt wurde, um die Übertragung von Dateien zu organisieren, die Teil des WWW sind.

Übrigens, wenn in der Adresszeile des Browsers nicht HTTP, sondern **HTTPS** steht, dann bedeutet das, dass der Datenaustausch verschlüsselt durchgeführt wird. Vielleicht erinnern Sie sich daran, dass Ihr Browser Sie mit einer Warnung genervt hat, wonach eine Webpage ein ungültiges Zertifikat genutzt hat. Ein solches Zertifikat ist bei der verschlüsselten Datenübertragung essentiell: Es ist der Schlüssel, mit dem Ihr Browser die Daten verschlüsselt, die an eine bestimmte Seite übertragen wird. In einer Veranstaltung zur Netzwerksicherheit werden Sie lernen, was dabei im Hintergrund passiert und warum Browser manchmal ein gültiges Zertifikat nicht anerkennen.

Wichtig: Das, was Sie hier kennen lernen gilt genauso für Anwendungen im WWW. Mit den selben Grundlagen, mit denen Sie eine Webanwendung entwickeln, können Sie also auch ein Spiel oder eine beliebige andere Anwendung entwickeln, die auf jedem Endgerät läuft, das vernetzt ist und auf dem ein Webbrowser läuft. Unterschiede beim Betriebssystem wie zwischen **Android** und **iOS** oder **Windows** und **Linux** sind dann vollkommen belanglos.

Kontrolle

- Sie wissen jetzt, was der Unterschied zwischen Internet und WWW ist.
- Sie wissen, dass eine Webanwendung lediglich eine Ansammlung von Dateien ist, die über das Internet auf Ihren Rechner übertragen werden.
- Sie wissen, dass es Standards für diese Übertragung gibt, die in Form sogenannter Protokolle veröffentlicht und genutzt werden.

- Sie wissen, dass diese Dateien in einer von vielen Sprachen verfasst sind, die als Markup Languages bezeichnet werden.
- Sie wissen, dass die beiden Buchstaben ML im Namen einer Sprache für Markup Language stehen können.

Ausblick

Jetzt werden Sie mehr über die Programmiersprachen erfahren, die neben Markup Languages bei Webanwendung und Webanwendungen zum Einsatz kommen.

5.2 Funktionalitäten unserer Webanwendung

Der folgende Satz aus der Architektur ist für uns als SoftwareentwicklerInnen essentiell:

Form follows function!

Das bedeutet, dass zuerst die Funktion definiert werden muss, bevor es um gestalterische Fragen geht. Aber wie Sie ohne schon erkennen konnten kommt vor der Funktion die Struktur. Und zur Funktion gehört dann noch die Speicherung und Wiederverwendung von Nutzerdaten. Im Gegensatz dazu konzentrieren sich MediendesignerInnen auf das Design.

Zusammengefasst folgt daraus für uns:

1. Zuerst legen wir die Struktur fest,
2. dann legen wir für jedes Element der Struktur die Funktion(en) fest, die dieses Element anbieten soll.
3. Abschließend legen wir fest, welche Nutzereingaben wie verarbeitet und ob sie gespeichert bzw. wiederverwendet werden sollen.

Dagegen gilt hier:

Design ist für uns irrelevant.

Wenn Sie sich dagegen mit Design beschäftigen wollen, dann wechseln Sie bitte zu einem Studium des Medien- und/oder Kommunikationsdesigns;

dort lernen Sie das, was Sie suchen. Hier sind Sie dann schlicht und ergreifend am falschen Platz.

Denn wir werden uns hier grundsätzlich mit dem Entwurf und der Entwicklung von Webanwendungen beschäftigen. Üblicherweise wird bei vergleichbaren Kursen und Tutorien im Netz damit begonnen, einen Webshop in HTML4.01 zu entwickeln, selbst wenn die Anbieter behaupten, es handle sich um eine Einführung in HTML5. Im Gegensatz dazu erhalten Sie hier einen ersten Einblick in aktuelle Methoden bei der Entwicklung von Webanwendungen.

5.2.1 Erste Schritte zur Webanwendung

Und der erste Schritt dazu hat etwas mit Stift und Papier zu tun:

1. Notieren Sie, was Ihre Webanwendung tun soll.
2. Streichen Sie alles durch, was angibt, wie sie es umsetzen soll.
3. Notieren Sie dann, welche Elemente Nutzer für welche Funktion angezeigt bekommen sollen. (Auch das Anzeigen von Texten ist eine Funktion.)
4. Streichen Sie alle konkreten Textentwürfe durch. („Guten Tag, lieber Nutzer“ ist ein konkreter Text, „Hier Begrüßung des Nutzers einblenden“ dagegen nicht.)
5. Streichen Sie alles durch, was definiert, wie diese Dinge angezeigt werden sollen. (Zur Erinnerung: Wir machen Medieninformatik, nicht Mediendesign.)

Wenn Sie jetzt gleich zum nächsten Abschnitt weitergehen, dann machen Sie etwas falsch, denn bevor Sie nicht ausführlich überlegt haben, was Sie auf Ihrer Seite unterbringen wollen, brauchen wir uns über die Programmierung oder das Design keine Gedanken zu machen.

5.2.2 Projektdokumentation und Arbeitsumgebung

Und damit sind wir beim wichtigsten Arbeitsmittel beim Projektstart: Papier. Für eigene Notizen im Format A4 oder A5, für Gruppenarbeiten sollten sie auch Bögen im Format A3 oder besser noch A2 besorgen. Klebeband, Textmarker und Stifte brauchen Sie natürlich auch. Schließlich wollen wir keine Papierflieger bauen.

Das mag in der Zeit von Digitalisierung und Vernetzung seltsam klingen, aber da Papier nun wirklich kein Luxusgut ist und es nunmal leichter ist, mehrere Blätter auf einen Tisch zu legen, um zu vergleichen, was den Mitgliedern eines Teams am besten gefällt, führt an dieser Stelle kein Weg an Stift und Papier vorbei. Idealerweise nutzen Sie dabei wenigstens ein Dutzend unterschiedlicher Farben. So können Sie beispielsweise jedem Mitglied Ihres Teams eine Farbe zuordnen. Dann ist erkennbar, wer welche Einträge vorgenommen hat.

Bei der Programmierung werden Sie noch mit einem Repository arbeiten, aber für den Moment bleiben wir bei Stift und Papier.

Für alle, die bereits mit Software gearbeitet haben, die eine solche Arbeit z.B. via Tablet ermöglicht und für diejenigen, die das gerne tun würden: Der einzige Grund, aus dem ich diese Methoden hier nicht unterstütze ist der, dass es heute noch keine günstigen Arbeitstische gibt, die diese Methoden für Teams unterstützen. Bei rund einhundert Studierenden pro Semester reden wir hier über wenigstens 10 entsprechende Tische, bzw. deutlich mehr als 100.000,- € zuzüglich der Kosten für zwei Räume à 50 m^2, in denen diese Tische stehen. Denn Sie werden schlicht und ergreifend eine große Arbeitsfläche benötigen; da stellen Tablets aufgrund Ihrer beschränkten Formate eine zu große Beschränkung dar. Ansonsten bin ich hier voll und ganz auf Ihrer Seite und hoffe, dass die nötige Hardware sobald als möglich zu bezahlbaren Preisen auf den Markt kommt. Sie wollen ein Beispiel für das sehen, was ich meine? Dann sehen Sie sich den Film “Casino Royale“ mit Daniel Craig an. Dort können Sie sehen, wie eine professionelle digitale und vernetzte Arbeitsumgebung für Teams aussehen kann. Allerdings fehlen dort Interfaces mit Tastatur und Maus, die fürs Programmieren unbedingt nötig sind.

5.2.3 Erste Übung

Die folgende Aufstellung ist für MS-Studierende gedacht, die in diesem Semester die Leistungsnachweise für PRG und Projekt 1 erwerben wollen. Für diese gilt:

Alle Schritte müssen in der genannten Reihenfolge bis zum 25. März 2016, 12.00 Uhr erledigt sein, sonst ist keine Teilnahme am Projekt 1 in diesem Semester mehr möglich.

Sollte die Veranstaltung in zwei Gruppen durchgeführt werden, dann gilt für die erste Gruppe die gerade genannte Frist, für die zweite Gruppe eine Frist bis zum 1. April 2016, 12.00 Uhr. (Nein, das ist kein Aprilscherz.)

1. Bevor Sie in einem Team mit einem Projekt beginnen, überlegen Sie für sich, was für ein Softwareprojekt Sie gerne umsetzen würden. Wichtig ist hier nicht, ob Sie sich vorstellen können, es umzusetzen, sondern dass Sie sich ein interaktives Programm überlegen, das Sie schon immer haben wollten.

   - Beschreiben Sie, was dieses Programm tut, bzw. wie Nutzer damit arbeiten, spielen oder wie auch immer interagieren.
   - Streichen Sie alles, was mit Design und konkreten Inhalten zu tun hat.
   - Es soll kein Roman dabei herauskommen: Kürzen Sie Sätze mit mehr als 10 Wörtern. Streichen Sie alle Adjektive. (Interessanterweise gibt es einen Literaturnobellpreisträger, der in seinen Romane die zweite dieser Vorgaben umgesetzt hat: Ernest Hemmingway)
   - Formulierungen wie „Es ist eine AAA-roguelike X4-Sim mit Shooter-RPG-Elementen.“ gehören ins Marketing und zeigen, dass Sie nicht im Stande sind auszudrücken, was Ihre Anwendung ausmacht.

2. Wenn Sie Ihre Projektidee in ein oder zwei Stunden wie gerade beschrieben grob skizziert haben, dann arbeiten Sie sich an Ihrem Rechner in Git ein. (Beachten Sie dabei alles, was im Kapitel „Vorbereitung fürs Programmieren“ steht.)

3. Erstellen Sie in Ihrem Repository ein Verzeichnis mit dem Namen Dokumentation. Speichern Sie darin Ihren Projektvorschlag in einer Datei mit dem Namen `Projektidee.txt` . Wenn Ihnen nichts eingefallen ist, dann tragen Sie das in der Datei ein.

4. Erstellen Sie (wenn noch nicht passiert) ein Repository bei GitHub, mergen Sie Ihr lokales Repository damit (bzw. pushen Sie eben Ihr lokales Repository auf GitHub) und tragen Sie mich als Collaborator ein. Mein Username bei GitHub lautet `MarkusAlpers` .

5. Melden Sie sich bei Helios für die Leistungsnachweise PRG und Projekt 1 an.
   Sollte die Anmeldung zurzeit nicht möglich sein, senden Sie mir bitte eine kurze Nachricht, damit ich in der Verwaltung alles nötige veranlassen kann.

6. Senden Sie mir an meine Hochschuladresse `markus.alpers@haw-hamburg.de` eine E-Mail **von Ihrer Hochschulmailadresse** (private Mailadressen werden nicht angenommen) mit folgenden Angaben:

- Als Bezug:
- `Anmeldung Projekt 1 (MS)`
- Als Inhalt die folgenden Zeilen (nur ausfüllen, also nicht explizit Nachname als Wort angeben, sondern Ihren Nachnamen usw. eintragen):
- `Nachname, Vorname, Matrikelnummer`
- `Ihren Usernamen bei GitHub`
- `Den Namen Ihres Repositories`

5.2.4 Mehr zu den Leistungsnachweisen für alle

Programmieren ist wie Fahrradfahren: Manche schaffen es auf Anhieb loszufahren und können relativ schnell an Wettrennen teilnehmen, andere brauchen Jahre, um auch nur den Einstieg zu schaffen. Es hat nichts mit auswendig lernen oder anderen Arten des „Lernens“ im Sinne von „büffeln“ zu tun, sondern beim Programmieren geht es (genau wie bei mathematischen Verfahren oder handwerklichen Tätigkeiten) fast ausschließlich darum, es wieder und wieder anzuwenden. Wenn Sie sich also nicht zu Hause hinsetzen und jede Woche mehrere Stunden selbst programmieren, dann ist Ihre Anwesenheit in diesem Kurs sinnlos und Sie werden keinesfalls einen Leistungsnachweis erhalten.

Umgekehrt hat Programmieren auf Hochschulniveau sehr viel mit Planung und Konzeption zu tun. Wenn Sie also die Inhalte der Veranstaltung ignorieren und einfach drauflos programmieren, wird auch das nicht funktionieren. Viele Studierende suchen dazu im Netz nach Lösungen, die sie nicht verstehen. Am Ende sagen Sie dann, dass Programmieren langweilig oder sinnlos ist. Dabei könnten Sie mit dem, was wir in dieser Veranstaltung besprechen praktisch alles selbst programmieren, was Sie an fertigen Programmen kaufen können.

5.2.5 Mehr zu den Leistungsnachweisen für MS-Studis

Wenn Sie einen Abschluss in Media Systems erwerben wollen, dann gibt es unter anderem die beiden Leistungsnachweise `Projekt 1` und `Einführung ins Programmieren`. Diese beiden Leistungsnachweise erhalten Sie, wenn Sie das Projekt 1 erfolgreich abgeschlossen haben. Nachdem Sie die eben genannten Bedingungen erfüllt haben wird die gesamte Kommunikation zum Projekt über Ihr Repository erfolgen. Der weitere Ablauf wird so erfolgen, dass ich Ihren individuellen Leistungsstand regelmäßig (mindestens alle vier Wochen einmal) anhand dessen kontrolliere, was im Repository steht und Ihnen dort individuelle Hinweise gebe,

wie Sie weiterarbeiten können.

Laut Modulhandbuch sollen Sie im Projekt nachweisen, dass Sie die Inhalte der Veranstaltung erfolgreich in einer Gruppe umgesetzt haben. Als Arbeitsaufwand sind 80 Stunden vorgesehen. Das schließt nicht die Vor- und Nachbearbeitung der Vorlesung bzw. des Seminars ein. Damit ergeben sich, dass Sie sich außerhalb der Veranstaltung rund 10 Stunden pro Woche mit den Themen der Veranstaltung und der Durchführung des Projekts beschäftigen müssen.

Häufig werde ich aufgefordert, Ihnen genau Angaben dazu zu machen, was Sie machen sollen. Das ist aber so konkret nicht möglich: Programmieren hat etwas mit der Fähigkeit zu tun, die eigenen Vorstellungen zu abstrahieren und sie in einer Form auszudrücken, die ein Computer ausführen kann. Ob Sie das schaffen lässt sich aber nicht daran messen, ob Sie die Umsetzung in fünf oder in 500 Zeilen durchführen. Es ist im Grunde so ähnlich wie mit Fahrradfahren: Wenn jemand mehrere hundert Meter mit einem Fahrrad ohne Stützräder gefahren ist, dann ist klar, dass er/sie es kann. Würde ich das aber auf den Leistungsnachweis übertragen, dann würden viele von Ihnen, die eine ausreichende Leistung erbracht haben keinen Leistungsnachweis erhalten. Und das wäre unnötig demotivierend. Deshalb lege ich keine solche messbare Grenze fest.

Es gibt aber noch einen weiteren Grund: Für den Leistungsnachweis müssen Sie eine gewisse Qualität erreichen, weil Sie MedieninformatikerInnen sind. Würde ich eine Liste aller möglichen Qualitätskriterien aufstellen, dann wäre das Ergebnis ein Katalog von mehreren hundert Seiten, den Sie erst dann verstehen würden, wenn Sie mehrere Jahre programmiert hätten. Einen Teil davon finden Sie allerdings in diesem Buch. Und selbst wenn Sie davon nur einen Teil umsetzen, stehen Ihre Chancen gut, den Leistungsnachweis zu erwerben.

5.2.6 Mehr zum Leistungsnachweis für MT-Studis

In Ihrem Studiengang gibt es für die Veranstaltungen `Programmieren 1` und `Programmieren 2` einen benoteten Leistungsnachweis, der nach Bestehen eines Projekts ausgestellt wird, das Sie in PRG2 durchführen. Dieses Projekt machen Sie voraussichtlich bei Herrn Wagener. Zur Durchführung kann ich deshalb nichts sagen. Bei mir erlernen Sie die Grundlagen, über die es eine schriftliche Prüfung am Ende dieses Semesters geben wird. Sie werden also anders als in der Schule nur eine Prüfung für die gesamte Veranstaltung schreiben.

5.3 Technische Grundlagen verteilter Anwendungen

Sie wissen bereits, dass eine verteilte Anwendung aus mehreren Programmen besteht, die über ein Netzwerk[1] miteinander kommunizieren. In den nachfolgenden Abschnitten und Kapiteln dieses Buches werden wir Schritt für Schritt alles besprechen, das Sie benötigen, um Sie Ihre erste verteilte Anwendung starten können: Eine Webanwendung, zu der es Nutzerkonten gibt und deren Inhalte sich abhängig von den Eingaben der Nutzer ändern können.

Wie Sie schon aus dem Abschnitt zur Vorbereitung auf die Entwicklung von verteilten Anwendungen wissen, gliedert sich dieser Teil des Buches in die folgenden Bereiche:

- Festlegung, welche Funktionalitäten unsere Webanwendung enthalten soll.
- Betrieb eines Webservers und Zugriff durch einen Webbrowser.
- Einführung in die Markup Language HTML, um statische Webanwendung zu entwickeln.
- Bereitstellung dieser Webanwendung auf dem Webserver.
- Erweiterung der Anwendung durch die Nutzung von PHP, sodass wir eine dynamische Webanwendung erhalten.
- Einbindung einer Datenbank, die auf dem Webserver bereit gestellt wird, um so die Speicherung von Daten über individuelle Nutzer zu realisieren.

Wenn Sie diese sechs Punkte beherrschen, dann verstehen Sie die grundsätzlichen Abläufe, die bei jeder verteilten Anwendung vorkommen. Der einzige Unterschied zwischen Ihnen und einem professionellen Entwickler solcher Anwendungen besteht in zwei Punkten: Zum einen die langjährige Erfahrung, zum anderen ein wesentlich besseres Verständnis, wie diese Abläufe stattfinden und wie hochkomplexe Projekte sinnvoll realisiert werden können.

5.4 Erste Gehversuche mit Server und Client

Nachdem wir die Planung der Webanwendung begonnen haben, fahren wir also damit fort, den Webserver zu nutzen bzw. zu prüfen, ob er soweit

[1] oder über ein Netzwerk von Netzwerken wie das beim „Internet“ der Fall ist

funktioniert, wie das für unsere Zwecke nötig ist:

Prüfen Sie bitte, ob EasyPHP den Server gestartet hat. Wenn das der Fall ist, öffnen Sie bitte einen Browser und geben dort in der Adresszeile (nicht im Suchfeld einer Suchmaschine!) das Wort localhost ein. Groß- und Kleinschreibung ist hier nicht von Belang.

Nun sollte eine Seite angezeigt werden, auf der am oberen Rand einige wenige Einträge mit Titeln wie „version“ erscheinen. Im Zentrum der Seite sollten drei Verzeichnisse angezeigt werden. Wenn Sie diese anwählen, werden Sie feststellen, dass Sie allesamt leer sind.

Wiederholen wir an dieser Stelle die Frage, was denn eigentlich der **Server** und was der **Client** ist (momentan haben wir jeweils nur einen): Der Server ist ein sogenannter Apache Server. Dabei handelt es sich um ein Programm, das mit Hilfe von Dateien, auf die es Zugriff hat, verschiedene Dateien generieren kann, die über ein beliebiges Netzwerk übertragen und von einem Browser als etwas angezeigt werden, das wir unter der Bezeichnung Webanwendung bzw. Webpage kennen. Das Programm, das also die Daten bzw. Dateien vom Server nutzt (eben der Client) ist dementsprechnd der Browser, den wir nutzen.

Wenn es keine Datei gibt, die eine Webanwendung generiert, dann erzeugt ein Webserver Daten, die vom Client so angezeigt werden, wie Sie das von Ihrem Rechner beim Dateibrowser kennen: Da werden einerseits Dateinamen mit Endungen und andererseits Verzeichnisnamen angezeigt. Und für jeden dieser Einträge gibt es dann noch ein Symbol. Aber auch hier haben wir es wieder mit HTML-Dateien zu tun, denn sonst könnte der Browser sie nicht anzeigen.

Aber selbst wenn der Server auf eine Datei Zugriff hat, die definiert, wie eine Webanwendung aussehen soll, bedeutet das nicht, dass der Server basierend auf dieser Datei eine Webanwendung-Ansicht generiert. Vielmehr müssen Sie ihn so konfigurieren, dass er das tut.

Für den Fall, dass Sie sich wundern: Die Bezeichnung **localhost** ist eine Art Alternativbezeichnung für eine bestimmte IP-Adresse. Darüber können wir einige Server nutzen, die auf unserem Rechner aktiv sind.

Was eine **IP-Adresse** ist? Ach ja richtig, Sie hatten ja noch keine Veranstaltung über Netzwerke... Eine IP-Adresse ist in Netzwerken (und damit auch im Internet) so etwas wie eine Telefonnummer für Computer. Diese IP-Adressen sind also eine Möglichkeit, damit Computer über Netzwerke Daten austauschen können. Denn ohne IP-Adresse hätten Sie ja kei-

ne Möglichkeit, einen bestimmten Rechner anzusprechen. Egal, ob Sie das wollen oder nicht: Dieser Datenaustausch passiert, wenn Sie einen Rechner, ein Smartphone oder welches vernetzte Gerät auch immer einschalten[2]. Im Gegensatz zu Telefonnummern sind IP-Adressen aber nicht automatisch fest einem Computer zugeordnet, sie sind also im Regelfall dynamisch zugeordnet.

Wenn sie nun im Netz surfen, dann geben Sie im Regelfall nicht die IP-Adresse eines Rechners ein, sondern den „Namen" des Standorts einer Webanwendung. Solche Namen haben wie alles einen Fachbegriff: Die **URL** (kurz für Unified Remote Location) einer Webanwendung der HAW Hamburg ist beispielsweise www.haw-hamburg.de wobei Sie im Regelfall das www. am Anfang weglassen können. Eine andere URL der HAW Hamburg lautet beispielsweise www.mt.haw-hamburg.de. Der Begriff URL ist zwar gebräuchlich, aber im Kern falsch, denn auch wenn das L für Location bzw. Standort steht, können Sie aus der URL nichts über den Standort eines Servers aussagen. Der einzige Standort, den Sie aus einer URL ablesen können ist der Standort des Verwaltungsgremiums, das für die Vergabe von URLs zuständig ist. (In Deutschland ist das die DENIC.) Präziser ist da die Bezeichnung **URI** (kurz für Unified Remote Identifier).

Leider gibt es keine unterschiedliche Bezeichnung für eine einzelne Ansicht einer Webanwendung oder die Gesamtheit aller Webanwendung, die unter einer URL zu finden sind. Sie müssen also jeweils überlegen, ob nun eine einzelne Seite oder die Gesamtheit aller Seiten einer Webanwendung gemeint ist.

Und jetzt kommt etwas, dass viele Menschen nicht wissen: Wenn Sie eine solche URI in einen Webbrowser eingeben, dann schlägt dieser in einem Verzeichnis nach, welche IP-Adresse zu dieser URL gehört. Ein solches Verzeichnis ist ein Programm (hier spricht man auch von einem **Dienst**), denn es muss ja im Stande sein, eine Antwort zu senden. Und wie nennen wir solche Programme? Richtig, es ist ein Server, der als Domain Name Server, kurz **DNS** Bezeichnet wird. Warum dieser Server so heißt, was Domänen sind, wie er funktioniert usw. usf. ist auch wieder Teil jeder Veranstaltung über Netzwerke. Ob ein Dienst nun ein einzelnes Programm oder eine verteilte Anwendung ist, soll uns an dieser Stelle nicht weiter interessieren. Der Begriff wird vorrangig im Bereich der **Nachrichten- und Kommunikationstechnik** verwendet, während wir als (Medien-)InformatikerInnen je nach Situation von Programm, verteilter Anwendung, Server, Client oder ähnlichem sprechen.

[2]Wie das im Detail funktioniert und wann genau eine Datenübertragung begonnen wird ist Teil jeder Veranstaltung zu IT-Netzwerken.

Wie bei allen Servern gilt auch hier: Dieser Server kann sich auf Ihrem Rechner befinden oder auf einem beliebigen Rechner irgendwo im Netz. Als Frau von der Leyen vor einigen Jahren forderte man müsse bestimmte Seiten im Netz sperren, indem man bei Suchanfragen ein Stoppschild einblenden würde, bewies Sie damit, dass sie nicht einmal die einfachsten Strukturen des Internet kannte. Denn um diese Stoppschilder zu realisieren, hätte jeder DNS-Server weltweit angepasst werden müssen. Und nur wer keine Ahnung vom Unterschied zwischen nationalem und internationalem Recht hat, kann auf die Überzeugung verfallen, dass beispielsweise die Administratoren eines DNS-Servers in Timbuktu sich bei ihrer Arbeit nach den Gesetzen der Bundesrepublik Deutschland richten. Vielleicht glaubt er oder sie auch an den Weihnachtsmann; wir beschäftigen uns hier mit der Informatik und ihren Auswirkungen in der Realität, deshalb wissen wir es besser.

Aber zurück zu unserem einfachen Webserver, bzw. zum Aufruf localhost im Webbrowser. Der Begriff localhost ist auch eine URI. Das bedeutet, dass ein Webbrowser im Regelfall auf einem DNS-Server nachsieht, welche IP-Adresse zu dieser Adresse gehört. Im Gegensatz zu anderen URIs gibt es aber für localhost eine Konvention: Dieser URL ist eine statische IP-Adresse zugeordnet: Es ist 127.0.0.1 Geben Sie diese Zahlenkombination einmal in Ihrem Browser in die Adresszeile ein.

Wenn Ihr Browser jetzt eine Übersicht von Internetseiten präsentiert, wie das bei Google der Fall ist, dann hat Ihr Browser entweder gar keine Adresszeile mehr, was leider immer öfter der Fall ist oder Sie haben die 127.0.0.1 in das Suchfenster eingetragen. Im zweiten Fall sollten Sie nochmal ernsthaft in sich gehen und überlegen, ob Sie nicht doch lieber irgend etwas anderes studieren wollen, denn dann haben Sie einen sehr hohen Nachholbedarf, was die Grundlagen der Bedienung eines Computers angeht.

Nochmal zur Erinnerung bezüglich der Begriffe statisch und dynamisch: Wenn wir in der Informatik davon reden, dass etwas statisch ist, dann handelt es sich dabei um eine Konvention, bzw. eine Vereinbarung und nicht um eine Art Naturgesetz. Denn im Gegensatz zu dem, was jemand mit dem Begriff „statisch“ verbinden könnte, sind alle Daten dynamisch: Ein Computer kann nur mit Werten arbeiten, die er in die Register seines Prozessors lädt. Und der Inhalt eines solchen Registers kann nicht statisch sein. Das wiederum entspricht auch letztlich jedem Naturgesetz: Das einzig konstante ist die Veränderung.

Wichtig: Diese Art der IP-Adressen ist als **IPv4** bekannt, wobei das v für Version steht. Der nächste genutzte Standard für IP-Adressen lautet **IPv6**

und wird immer häufiger eingesetzt. Bei Webanwendungen ist aber zurzeit IPv4 noch Standard. Die Unterschiede zwischen den Versionen 4 und 6 sind auch wieder Teil jeder Veranstaltung zu IT-Netzwerken.

Kontrolle

- Sie wissen, wie Sie auf Ihren Rechner einen Webserver starten und Sie wissen auch, warum hier nicht automatisch eine Webanwendung erzeugt wird.
- Sie wissen, dass auf Ihrem Rechner sowohl Client als auch Server laufen können und Sie wissen, wie man diese Art von Server bzw. Client bei Webanwendungen nennt, bzw. welche Programme Sie dafür nutzen.
- Sie wissen, wie Sie auf die Einstiegsseite kommen, die der Webserver generiert.
- Sie kennen auch die Aufgabe eines DNS Servers, wissen was die Abkürzungen IP und URL bzw. URI bedeuten und wie diese zusammenhängen oder eben nicht zusammenhängen.
  (Anmerkung: Die Bezeichnung DNS Server ist im Grunde doppelt gemoppelt, denn das S in DNS steht ja bereits für Server, aber dennoch ist es üblich, diese Server so zu nennen.)

5.4.1 Eine erste Webanwendung mit PHP

Erstellen Sie bitte das folgende Programm mit Hilfe eines Editors und speichern Sie es im Verzeichnis `C:/ProgramFiles(x86)/EasyPHP-DevServer-14.1VC11/data/localweb` unter dem Namen `phpinfo.php` . Wenn Sie nicht EasyPHP installiert haben oder das Verzeichnis von EasyPHP bei der Installation geändert haben, müssen Sie prüfen, wo Sie die Datei speichern müssen. Es ist auch möglich, dass der Name des Verzeichnisses sich bei einer späteren Version von EasyPHP ändern wird.

```
<html>
<body>
<?php
phpinfo();
?>
</body>
</html>
```

Rufen Sie nun die Webanwendung auf. In der Ansicht sollte neben den drei Verzeichnissen auch der Dateiname phpinfo.php erscheinen. (Sonst haben Sie beim Speichern einen Fehler gemacht.) Wählen Sie diesen Dateinamen einmal mit der Maus an, so wie Sie es bei einem beliebigen Link auf einer Webanwendung tun würden.

Sie erinnern sich an die Aussage vom Anfang, wonach die Entwicklung verteilter Anwendungen umfangreiche Kenntnisse aus vielen Bereichen benötigt? Die Seite, die Ihnen jetzt angezeigt wird, zeigt Ihnen die Konfiguration (also die Einstellung) aller Komponenten an, die alleine bezüglich der Darstellung von Elementen auf einer Webanwendung relevant sind. Das ist aber nur ein Teilbereich aller Konfigurationen, die bei der Entwicklung und beim Betrieb einer Webanwendung oder einer anderen verteilten Anwendung relevant sind. Und? Sind Sie schockiert über die Masse an Dingen, die Sie nicht verstehen? Gut, denn dann werden Sie hoffentlich nicht so überheblich wie die breite Masse an Entwicklern, die glauben, dass Sie Meister des Netzes sind, nur weil sie im Stande sind, eine Webanwendung zu entwickeln und zu betreiben. Denn dafür brauchen Sie so gut wie nichts von dem zu verstehen, was Ihnen gerade angezeigt wird. Aber bedenken Sie: All diese Einstellungen sind wichtig und müssen richtig konfiguriert sein, damit Ihre Webanwendung richtig funktioniert und um es Angreifern so schwer wie möglich zu machen, Ihre Webanwendung zu manipulieren.

Kontrolle

Sie wissen jetzt grundsätzlich, wie Sie eine Webanwendung auf einen Server übertragen können, und dass diese Übertragung bereits genügt, damit diese von Nutzern aufgerufen werden kann.

Sie wissen jedoch bislang nichts darüber, wie Sie eine Webanwendung programmieren müssen, um eigene Inhalte darauf zu präsentieren. Wir haben auch noch nicht über mögliche rechtliche Folgen einer Veröffentlichung von Inhalten auf einer Webanwendung gesprochen oder darüber, welche rechtliche Folgen es haben kann, dass Sie überhaupt eine Webanwendung ins Netz stellen. Doch bevor wir über weitere Details der Programmierung von Webanwendung sprechen, sollten wir dafür fünf Minuten aufwänden.

Ausblick

Nachdem Sie jetzt die Voraussetzungen geschaffen haben, um auf Ihrem System Webanwendungen zu entwickeln, wenden wir uns den Details der Softwareentwicklung in den drei Bereichen zu, mit denen Sie zu tun haben werden: Modell, Steuerung und langfristige Speicherung. Für jeden dieser

drei Teile verwenden wir hier eine eigene Programmiersprache: Für das Modell HTML in der Version 5, für die Steuerung PHP in der Version 5.6 und für die langfristige Speicherung MySQL in der Version 5.6. An dieser grundsätzlichen Aufteilung und der konzeptionellen Arbeit wird sich bei Webanwendungen nichts ändern, auch wenn Sie z.B. JavaScript anstelle von PHP oder MongoDB anstelle von MySQL verwenden.

Seit einigen Monaten ist PHP 7 auf dem Markt. Wenn Sie damit weiterarbeiten wollen, sollten Sie sich hier einarbeiten. Da ich mich zurzeit darum kümmere, dieses Buch fertig zu stellen habe ich mir die Änderungen noch nicht im Detail angesehen und werde das auch bis Ende des Semesters nicht tun.

5.5 Programmierung von Webanwendungen

Die bekannteste Markup Language ist **HTML**, die HyperText Markup Language. Seit dem Herbst 1999 wurde HTML in der Version 4.01 verwendet, seit Herbst 2014 steht HTML5 zur Verfügung. Beide Versionen haben nur gemein, dass HTML5-Dokumente auch HTML4-Code beinhalten können, und dass dieser wie bisher ausgeführt wird. Sonst nichts. Wer denkt, HTML5 sei nur eine Erweiterung von 4.01, hat es nicht verstanden. Es ist im Kern eine vollständig neue ML, die entwickelt wurde, um alle nur denkbaren multimedialen und auch interaktiven Inhalte verfügbar zu machen.

Die Kompatibilität zu Version 4 wurde eingeführt, damit es keine Probleme mit alten Webanwendung gibt. Insbesondere unterstützt HTML5 direkt die objektorientierte Softwareentwicklung und ist darauf ausgelegt im Verbund mit JavaScript jede Art von Programmen im Browser zu realisieren. Das bedeutet unter anderem, dass **Flash** bzw. ActionScript nunmehr überflüssig geworden ist. **ActionScript** ist die Sprache, in der Flash-Anwendungen programmiert werden und die zum Eigentum von Adobe gehört. Es ist eine Alternative zu JavaScript.

Wie gesagt definieren Sie in einer Markup Language lediglich Elemente bzw. Container, die Texte und Verweise auf Dateien enthalten. Die Darstellung des Inhalts dieser Container wird dagegen über sogenannte Cascading Style Sheets (kurz **CSS**) festgelegt und über die Einstellungen des Browsers auf dem Rechner eines Nutzers.

An dieser Stelle ein Hinweis für diejenigen von Ihnen, die bereits mit HTML programmiert haben: (Alle anderen lesen bitte beim nächsten Absatz weiter, da Sie das hier noch nicht verstehen können.) Sie haben wahrscheinlich

schon Attribute wie align genutzt. Das dürfen Sie unter HTML5 nicht mehr in dieser Form tun! Hier müssen Sie alles, was sich auf das Layout bezieht in CSS programmieren. Das mag ungewohnt sein, ist aber sinnvoll, weil auf diese Weise die Trennung zwischen der Definition von Elementen (was unter HTML passiert) und ihrer Darstellung (was unter CSS programmiert wird) eindeutig geklärt ist.

Um nun Programme wie beispielsweise Spiele im Browser zu entwickeln werden verschiedene Ansätze gewählt. Auch hier wurde durch die Einführung von HTML5 ein Standard eingeführt: Während bei Version 4 keine Programmiersprache als Standard vorgesehen ist, sieht Version 5 die Anbindung von Programmen in der Programmiersprache JavaScript als Standard vor. JavaScript ist eine der mächtigsten Sprachen, die Sie zurzeit nutzen können, um Anwendungen auf einem Rechner oder in einem Browser zu entwickeln. Auch aus diesem Grund werden Sie JavaScript immer öfter auf Webanwendung finden.

Eine Sprache, die früher häufig eingesetzt wurde und deshalb auch heute noch weit verbreitet ist, ist **PHP**. JavaScript ist zwar mächtiger und aus diesem Grund empfehlenswerter, aber es hat einen Nachteil für Sie, wenn Sie sich in die professionelle Teamarbeit einarbeiten wollen: Während Sie in PHP nichts programmieren können, das Sie in HTML oder CSS programmieren können, ist es möglich, nahezu eine vollständige Webanwendung in JavaScript zu entwickeln. Das führt dann bei Einsteigern schnell dazu, dass der JavaScript-Programmierer Teile übernimmt, die andere umsetzen sollten. Und die haben dann nichts mehr zu tun.

Ursprünglich wollten die Entwickler durch den Einsatz von Sprachen wie PHP und JavaScript nur erreichen, dass Webanwendung dynamisch werden. Eine **dynamische Webanwendung** ist schlicht eine Webanwendung, deren Inhalte sich z.B. durch Eingaben von Nutzern ändern können. Die Möglichkeiten, die HTML5 zusammen mit JavaScript bietet gehen weit darüber hinaus.

Zu guter Letzt brauchen Sie als Entwickler einer Webanwendung noch eine Möglichkeit, um Daten langfristig zu speichern. Hier kommen die sogenannten **Datenbanken** zum Einsatz: Eine Datenbank ist eine Ansammlung von Tabellen, in denen Daten in standardisierter Form abgespeichert werden. Im Gegensatz zu Dateien liegen Datenbanken dabei ständig im Speicher eines Rechners vor und können deshalb genutzt werden, ohne dass Sie zunächst von einer Festplatte oder einem USB-Stick geladen werden müssten.

Auch hier bekommen wir es mit Programmiersprachen zu tun, namentlich

mit den sogenannten Query Languages (kurz QL) oder Structured Query Languages (kurz **SQL**), was übersetzt so viel wie strukturierte Anfrage-Sprache bedeutet. Denn nichts anderes tun diese Sprachen: Sie ermöglichen es einem Nutzer in standardisierter Form Anfragen an eine Datenbank zu stellen und geben die Antwort der Datenbank in einer Form aus, die für Menschen lesbar ist. Die bekannteste dieser Sprachen ist **MySQL**.

Wichtig: SQL steht dagegen nicht für sequel (zu Deutsch: Nachfolger). Es ist aber im englischsprachigen Raum durchaus üblich, Abkürzungen nicht zu buchstabieren, sondern sie wie ein Wort auszusprechen. Deshalb wird SQL gelegentlich als sequel bezeichnet.

Während PHP so entwickelt wurde, dass es die Nutzung von MySQL direkt unterstützt, benötigt JavaScript eine Erweiterung dafür. Eine recht neue Erweiterung heißt **Node.js**. Dieser Ansatz eine Sprache über Erweiterungen zu ergänzen, anstatt diese von vornherein in die Sprache zu integrieren, wirkt auf den ersten Blick komplizierter als die vermeintlich einfache Kombination von PHP und MySQL, aber es ist einfach nur ein Ansatz, der es ermöglicht, die Sprache selbst nicht zu umfangreich werden zu lassen. Was das im Detail bedeutet können Sie in Veranstaltungen zum Software Engineering lernen.

5.5.1 MVC – Das Model View Controller Pattern

Diese Aufteilung der Programmierung einer Webanwendung entspricht einem Entwurfs- und Architekturmuster, das auch in der Wirtschaft angewendet wird. Dieses Muster wird als MVC (kurz für Model View Controller) bezeichnet. Pattern ist schlicht das englische Wort für Muster.

Solche Modelle spielen bei der Entwicklung großer Softwareprojekte eine essentielle Rolle, denn indem wir sie nicht wie einen Monolithen, sondern als Kombination von einzelnen Komponenten entwickeln, können wir leichter Fehlerkorrekturen durchführen und Änderungen einbauen.
Außerdem können wir so die Arbeit leicht innerhalb eines Teams mit mehreren Dutzend Entwicklern verteilen.

Allerdings kommt es nur sehr selten vor, dass für jeden Teil eines Patterns auch eine eigenständige Programmiersprache genutzt wird. Das macht es dann gerade für Einsteiger schwer, zu verstehen, wo die Grenze zwischen Model und View und Controller liegt. Das ist bei der Kombination aus HTML5, CSS und PHP oder JavaScript anders: Hier können wir jedem Bestandteil des Patterns genau eine Sprache zuordnen, so wie das oben schon passiert ist. Die SQ-Sprache lassen wir hier mal außen vor; sie dient ja nur dazu, die langfristige Speicherung von Daten zu ermöglichen.

In unserem Fall ist HTML5 die Sprache für das Model, CSS ist die Sprache für den View und PHP bzw. JavaScript ist die Sprache für den Controller. Deshalb werden wir uns im Kapitel über HTML auch nur darum kümmern festzulegen, aus welchen Teilen wir eine Webanwendung zusammensetzen können. Aus dem gleichen Grund werden wir uns im Kapitel über CSS ausschließlich darum kümmern, wie die Elemente der Seite dargestellt werden sollen. Und im Kapitel über PHP bzw. JavaScript wird es dann ausschließlich darum gehen, wie wir Änderungen steuern können, die während der Nutzung der Webanwendung auftreten.

Diejenigen von Ihnen, die Media Systems studieren werden später in der Veranstaltung Software Engineering eine Vielzahl von Patterns kennen lernen. Leider bleibt dort kaum Zeit, sich einem dieser Design Patterns ausführlich zu widmen. Aber so haben Sie jetzt schon die Möglichkeit, den Nutzen eines solchen Patterns kennen zu lernen.

Kontrolle

- Sie wissen, dass Sie bei der Programmierung von Webanwendung vier Arten von Programmiersprachen nutzen, und dass Sie damit ein Design Pattern der Informatik umsetzen, den MVC:
  - Markup Languages dienen dazu, Container zu definieren, die die Inhalte Ihrer Webanwendung enthalten oder die auf Dateien verweisen, die als Teil einer Webanwendung angezeigt werden.
  - CSS dient dazu, die Darstellung der Inhalte zu gestalten.
  - Sprachen wie JavaScript und PHP werden dazu genutzt, um dynamische Webanwendung bzw. Webanwendungen zu realisieren.
  - SQL-Sprachen dienen dann dazu, um z.B. Nutzereingaben dauerhaft zu speichern, um Kataloge von Waren bereitzustellen oder andere große Mengen an Daten aufzubewahren.
- Ihnen ist klar, dass Sie damit wesentlich mehr realisieren können als „nur“ Webpages, auch wenn Sie noch nicht wissen, wie Sie das tun können. Ausblick

Jetzt werden Sie den nächsten Schritt der Web-Evolution kennen lernen: Das semantische Web.

5.6 Das semantische Web

Bis hierher haben Sie nur über Dinge gelesen, die Sie unter Umständen schon wussten. Doch nun kommen wir zu einem Thema, dass den meisten Menschen und leider auch vielen Webentwicklern nicht bekannt ist, obwohl es der nächste Schritt für die Nutzung des WWW ist. Die Rede ist vom semantischen Web. Ohne das Verständnis, was das semantische Web ist, können Sie mit HTML5, dem neuen Standard für Webanwendung nichts anfangen. Sehen Sie sich dazu bitte die folgenden Einleitung ein: `https://plus.google.com/+ManuSporny/posts/FPwMGWhgQYh?cfem=1` Sehen wir uns das mal im Detail an:

5.6.1 Syntax und Semantik

Den einen dieser Begriffe haben Sie wahrscheinlich in der Schule kennen und hassen gelernt. Dabei stehen die beiden für ein ausgesprochen sinnvolles Konzept. Mit **Syntax** werden all die Aspekte einer Sprache bezeichnet, bei denen es darum geht, welche Buchstaben und andere Zeichen in welcher Reihenfolge notiert werden dürfen. (Denken Sie an so etwas wie Satzbau, Kommasetzung, usw.) Wenn also eine Sprache ohne Syntax auskommen könnte, gäbe es keine Absprache darüber, welche Wörter es gäbe und wie ein Satz aussehen kann. Und egal ob Sie diesen Stil nun mögen oder nicht, auch der folgende Satz folgt einer Syntax: „Ey Alter, was geht'n?“

Damit kommen wir zur **Semantik**. Mit Semantik bezeichnen wir alles, was uns sagt, welche Bedeutung etwas in einer Sprache hat. Es ist wichtig zu verstehen, dass zwei syntaktisch unterschiedliche Sätze in semantischer Hinsicht gleich sein können. So gibt es einen großen syntaktischen Unterschied zwischen „Ey Alter, was geht'n?“ und „Hallo Herr Schulz, wollen wir Essen gehen?“ aber semantisch ist dieser Unterschied eher gering. Das gilt genauso für Programmiersprachen: Vieles lässt sich in gänzlich unterschiedlicher syntaktischer Form programmieren und erfüllt doch die selbe Aufgabe.

Das führt uns auch nochmal zu der Antwort auf die Frage, warum bei es bei den Projekten der MS-Studierenden keine feste Vorgabe für den Umfang gibt: Da mit extrem unterschiedlichem Code identische Funktionalitäten zu realisieren sind und es nicht um die Form, sondern um die Funktion geht, sind Sie in der Wahl der Form recht frei. Da aber auch dir Funktion von Element zu Element unterschiedlich ist, sind Sie auch in der Wahl der Formen recht frei. Nur die Programmiersprachen und die zu verwendende Version sind vorgegeben.

Ein fähiger Softwareentwickler ist somit wie ein Diplomat oder ein Dichter: Er kann mit Sprachen umgehen wie ein Küstler. Und das nicht, weil er ihre Regeln gelernt hat, sondern weil er sie nutzt, um Ideen und Konzepte in eleganter Form auszudrücken. Deshalb ist der erste Schritt zu einem guten Programm auch nicht der Griff zur Tastatur, sondern zu Stift und Papier, um Konzepte und Relationen zu skizzieren, sie zu überarbeiten und ein sinnvolles Ganzes daraus zu gestalten. Erst wenn dieser Schritt abgeschlossen ist, beginnt die eigentliche Programmierung.

Wieder zurück zum Begriff der Semantik: Es gibt einen sehr großen Unterschied zwischen der Semantik bei gesprochenen Sprachen und bei Programmiersprachen: Die Semantik einer Zeile einer Programmiersprache ist immer eindeutig: Sie besagt je nach gewähltem Paradigma entweder was der Computer tun soll oder wie er es tun soll. Bis auf HTML5 gibt es aber noch keine Programmiersprache, die auch aussagt, was das ganze im Sinne der Kommunikation von Menschen bedeutet!

Stellen Sie sich dazu die folgende Situation vor, die Schultz von Thun in seinem Standardwerk über zwischenmenschliche Kommunikation präsentiert: Ein Mann sitzt am Steuer seines Wagens und seine Frau sagt zu ihm: „Es ist grün.“ Dieser Satz lässt sich unterschiedlich interpretieren. Das ist das Problem mit der Semantik in gesprochenen Sprachen: Sie ist im Regelfall nicht eindeutig.

Die meisten Anfänger einer Programmiersprache verstehen deshalb nicht, dass es bei Programmiersprachen keine unterschiedliche Interpretation eines Programms gibt. Wenn es also bei einer Programmiersprache das Äquivalent zu „Es ist grün.“ gibt, dann hat dieser Programmbefehl genau eine Bedeutung. Daraus folgt auch, dass Informatiker in aller Regel versuchen, Dinge so eindeutig wie möglich zu formulieren. Es geht ihnen nicht darum, Dinge zu verkomplizieren, sondern darum, Missverständnisse zu vermeiden.

Woran Sie erkennen können, dass selbst erfahrene Programmierer häufig ignorieren, dass die Semantik von Programmiersprachen eindeutig ist? Ganz einfach: Immer wenn Sie den Satz „Warum macht der das denn nicht?!“ hören, ignoriert jemand diese fundamentale Tatsache. Aber da Programmierer eben auch nur Menschen sind, ist das vollkommen in Ordnung.

Aber was hat das mit dem WWW zu tun? Und wie passt hier der Begriff semantic web ins Bild? Dazu eine kleine Übung:

Aufgabe

- Suchen Sie im Internet (per google.de und duckduckgo.com) nach dem Wort Musik. Vergleichen Sie die Ausgaben der beiden Suchmaschinen.
- Überlegen Sie, warum Menschen mit den Ergebnissen dieser Suche unzufrieden sein könnten, egal welche Suchmaschine sie nutzen.
- Überlegen Sie, was Sie eingeben müssten, damit eine Suchmaschine Ihnen Angebote für CDs Ihrer Lieblingsband anzeigt.

Kontrolle

Sie haben eine erste Idee davon, dass das semantic Web etwas damit zu tun, welche Bedeutung Wörter und multimediale Inhalte im WWW haben.

5.6.2 Semantik und das WWW

Wichtig: Oben haben Sie eine Bedeutung des Wortes Semantik erfahren: Es ist die Bedeutung eines Programmbefehls, bzw. eines Teils eines Computerprogramms. Beim semantic Web geht es aber nicht darum, wie ein Computerprogramm auszuführen ist. Wenn Sie das verwirrt, denken Sie bitte daran, dass der Begriff der Semantik allgemein als „Bedeutung und Interpretation von etwas“ übersetzt werden kann. Wenn wir also vom semantic Web reden, dann geht es um die Bedeutung von etwas, das mit dem WWW zu tun hat, bzw. Teil des WWW ist.

Zurück zum eigentlichen Text:

Die Aufgabe oben war schwierig, aber durch die einleitenden Erklärungen zum WWW hatten Sie das Wissen, um die Ursache des Problems zu erkennen. Dort haben Sie erfahren, dass Webanwendungen in Markup Languages programmiert werden. Sie haben dort ebenfalls erfahren, dass Sie in diesen Sprachen lediglich Container definieren können. Und was ist das Problem mit Containern? Genau: Die meisten Menschen sehen nur die Verpackung aber nicht den Inhalt. Und das gleiche gilt für Container einer Markup Language: Suchmaschinen tun sich schwer damit, die Webanwendung zu finden, nach denen ein Nutzer sucht, weil sie keine semantischen, sondern ausschließlich syntaktische Informationen bekommen, wenn ein Benutzer eine Eingabe macht. Und umgekehrt stellen Webanwendung ebenfalls keine semantischen, sondern ausschließlich syntaktische Informationen zur Verfügung.

Es hat seit der Einführung von HTML4.01 mehrere Versuche gegeben, dieses Problem zu lösen. Ein Lösungsansatz besteht darin, sogenannte Labels zu programmieren. Damit befestigt der Entwickler einer Webanwendung

gewissermaßen Schilden an die Container. Aber auch das ist ja wieder nur eine syntaktische Information, also löst es das Problem nicht.

Kontrolle

Wenn Computer Dateien in einer ML per HTTP austauschen, dass haben Sie nicht die geringste Ahnung, womit sie es zu tun haben. Also müssen wir einen Standard haben, mit dem wir semantische Informationen in ML-Dateien unterbringen können. Denn nur dann können die Computer nach diesen Informationen suchen.

5.6.3 DOM und Microdata

Wenn Sie sich schon einmal mit der Entwicklung von Webanwendung beschäftigt haben, dann haben Sie vielleicht schon die Abkürzung DOM gesehen. Das Document Object Model (kurz DOM) ist die Basis für die Nutzung von JavaScript im Browser. Es geht hier schlicht darum, dass jeder Container einer ML von der Sprache JavaScript als ein eigenständiges Objekt behandelt werden kann. Die Sprache kann dann den Inhalt eines solchen Containers ändern. Über DOMs kann außerdem definiert werden, wie verschiedene Container voneinander abhängen. All das hilft uns aber bei der Suche nach einem semantic Web nicht weiter, weil es immer noch keine semantischen Informationen bereitstellt, sondern nur klärt, wie die Elemente einer Webanwendung voneinander abhängen.

Microdata sind nun gewissermaßen eine Erweiterung von DOMs, die mit HTML5 eingeführt wurden: Hier enthält ein Container Einträge, die in standardisierter Form angeben, welche Bedeutung ein Eintrag in einem Container hat. Wenn Sie also Microdata verwenden, dann bedeutet das, dass Sie eine Anschrift in HTML5 so einprogrammieren, dass der Browser erkennen kann, dass es eine Anschrift ist, welcher Teil die Straße ist, welche Textpassage der Name ist, usw. Ein Besucher dieser Webanwendung braucht dann nichts weiter zu tun, wenn er diese Anschrift in sein Adressbuch übernehmen will oder die Anschrift in einem Kartenprogramm suchen will: Durch die Microdata kann der Browser die entsprechenden Verlinkungen erzeugen und die nötigen Daten weitergeben. Mit Microdata erhalten wir also eine semantische Webanwendung.

Bitte denken sie nicht, dass das nur für Anschriften gilt. Auf der Seite `http://www.schema.org/docs/full.html` finden Sie einen Überblick über alle Arten von Microdata, die Sie verwenden können. Bitte versuchen Sie nicht, all diese Typen auswendig zu lernen; wie so oft in der Informatik ist auswendig lernen unsinnig, da kontinuierlich Änderungen erfolgen und es viel wichtiger ist, zu wissen, wo Sie etwas finden, als dass

Sie wissen, wie es im Detail aussieht.

Neben naheliegenden Inhalten wie Anschriften finden Sie hier auch Microdata für Veranstaltungen. Richtig gelesen: Wenn Sie innerhalb eines Containers einen Eintrag als solch einen Typ von Microdata programmieren, dann brauchen Sie auf der ganzen Seite nicht ein einziges Mal den Begriff Veranstaltung zu benutzen: Eine Suchmaschine kann alleine durch diese Typangabe erkennen, dass es sich um eine Veranstaltung handelt und nicht etwa um einen Leitfaden für die Planung und Durchführung einer solchen.

Neben den negativen Möglichkeiten sorgt das semantic Web auch dafür, dass Manipulationen durch Betreiber von Preisportalen und ähnlichen Webanwendung in Zukunft weniger erfolgreich sein werden: Da bei konsequenter Anwendung von HTML5 und Microdata eine Suchmaschine direkt erkennt, wo eine Seite ein Angebot z.B. für Flugreisen anbietet, werden Nutzer langfristig eine eigene Suche nach Flugreisen durchführen können. Deshalb werden dürften diese Portale mittelfristig wieder vom Markt verschwinden.

Und das Großartige bei all dem ist, dass es wie beim WWW kein gewinnorientierte Unternehmen gibt, dass für die Nutzung von HTML5 Geld verlangt.

Kontrolle

- Sie wissen jetzt, dass Sie in HTML5 eine fast schon unüberschaubare Menge an Informationen so verpacken können, dass Browser wissen, um was für Daten es sich handelt.
- Sie haben eine erste Vorstellung davon, wie umfangreich die Auswirkung des semantic Web ist.

5.6.4 Zusammenfassung

Sie haben jetzt einen ersten Überblick über Techniken und Technologien, die Sie nutzen können, um Webanwendungen entwickeln können. Wenn Sie dieser Einleitung aufmerksam gefolgt sind, dann haben Sie außerdem verstanden, dass es bei der Entwicklung einer Anwendung, eines Computerprogramms oder eine verteilten Anwendung nicht zuerst darauf ankommt, welche Programmiersprache Sie nutzen wollen, sondern welche Konzepte und Ideen Sie umsetzen wollen.

5.7 Übertragung der Dateien auf einen Webserver im Netz

Wenn Sie im Gegensatz zum hier beschriebenen Vorgehen bereits einen Webserver nutzen, der Ihre Webanwendung im Netz anbietet und über das Internet erreichbar ist, dann müssen Sie noch wissen, was für ein Programm Sie zur Datenübertragung nutzen können.

Hierzu müssen Sie zunächst wissen, was ein **Protokoll** ist und welche Protokolle es gibt.

Protokolle sind Vereinbarungen darüber, wie bestimmte Abläufe geregelt werden. Klingt kompliziert? Ist es nicht. Ein praktisches Protokoll ist die Begrüßung: Es kann sein, dass Sie jemandem zur Begrüßung die Hand schütteln, ihr ein einfaches Hallo sagen, oder wie auch immer Sie die Begrüßung handhaben. Wenn Sie mit einer anderen Person eine feste Vereinbarung treffen sollten, wie die Begrüßung ablaufen soll, dann entspricht das dem, was wir als Protokoll in einem Netzwerk bezeichnen. Sie habens noch nicht verstanden? Dann sehen Sie ein paar Folgen Star Trek.

Wie Sie bereits wissen, gibt es verschiedene Arten, wie man Daten darstellen kann. Zur Erinnerung: In der **Nachrichtentechnik** wird das über die sogenannte **Codierung** festgelegt. In der Informatik werden üblicherweise nur Codes genutzt, die sich als Codetabelle darstellen lassen oder die mittels einer mathematischen Funktion erfolgen. Vereinfacht ausgedrückt definiert ein Protokoll unter anderem, mittels welcher Codierung Daten zwischen Server und Client ausgetauscht werden.

Ein Protokoll bekommen Sie fast immer angezeigt, wenn Sie im Netz unterwegs sind. Es handelt sich um das **HTTP**, das Hyper Text Transfer Protokoll. Übersetzt ist das also ein Protokoll, das festlegt, wie Hypertexte (was auch immer das sein mag) von einem Client angefordert und an ihn übertragen werden.

Nun also zum Begriff **Hypertext**: Wie Sie wissen werden in Webanwendungen Inhalte angezeigt, von denen aus Sie über Links (Fachbegriff **Hyperlink**s) zu anderen Webanwendungen bzw. zu anderen Inhalten gelangen. Um zu betonen, dass ein Dokument aus Texten und aus Verbindungen zwischen entfernten Textpassagen besteht, wurde der Begriff des Hypertexts entwickelt. (Das ist heute eine Selbstverständlichkeit, aber versuchen Sie mal einen Link jemandem zu erklären, der bislang nur Bücher aber keine Webpages kennt. Denn bei der Nutzung eines Links springen Sie ja direkt an eine bestimmte Stelle, ohne erst nachschlagen und nach der entspre-

chenden Stelle im Buch suchen zu müssen.)

Nun aber zu zwei Protokollen, die Sie für die Datenübertragung auf einen Webserver kennen müssen: Das FTP und den SSL. Wenn Sie konzentriert gelesen haben, dann haben Sie bemerkt, dass HTTP es nicht ermöglicht, ganze Dateien beliebiger Dateiformate zu übertragen, sondern dass es nur dazu dient, die Inhalte von Webanwendungen anzufordern und zu übertragen.

Das **FTP**, kurz für File-Transfer-Protokoll löst nun genau diese Aufgabe: Es definiert, wie ein Client Dateien auf einen Server übertragen kann. Aus Sicherheitsgründen sollten Sie jedoch prüfen, ob dieses Protokoll für Ihre Zwecke ausreichend ist. Denn eine Datenübertragung per FTP ist öffentlich. Das bedeutet in Kurzform: Alle angeschlossenen Nutzer können sämtliche Daten kopieren, die Sie über ein Netzwerk übertragen. Ja! Natürlich gilt das auch für Ihre Passwörter. Und das gleiche Prinzip gilt auch für HTTP. Denn diese Dienste wurden entwickelt, um bestimmte Daten mit einer möglichst hohen Effizienz über Netzwerke zu übertragen und jede Form von Sicherheit reduziert im Regelfall die Effizienz einer Übertragung. In den Äußerungen mancher politisch interessierten Personen zeigt sich hier das mangelnde Verständnis für die Technologie: Denn das hat nichts mit einem Wunsch nach allumfassender Überwachung zu tun, das hat etwas mit der Anwendung von Naturgesetzen zu tun.

Wollen Sie dagegen weitestgehend sicherstellen, dass niemand die übertragenen Dateien ändern oder auch nur kopieren kann, dann sollten Sie eine Verbindung mittels **SSL**, kurz für Secure-Socket-Link wählen. Allerdings kann es sein, dass der Webserver diese Art des Datentransfers nicht unterstützt. Gerade bei Billigangeboten wird SSL-Unterstützung gerne als kostenpflichtiges Update verkauft. Ob SSL nun ein eigenes Protokol oder die Erweiterung eines Protokolls ist, lassen wir hier einmal außen vor.

Ähnlich wie SSL bietet auch **HTTPS** eine sicherere Möglichkeit, um im Netz Daten zu übertragen. Beachten Sie aber bitte: Eine absolute Sicherheit gibt es auch in Netzwerken nicht. Der Sinn von Netzwerken bestand sehr lange auch ausschließlich darin, Verbindungen zu ermöglichen, nicht darin, die Vertraulichkeit der übertragenen Daten sicherzustellen. Über Details zu diesen Themen hören Sie mehr in der Veranstaltung Netzwerke und Internetsicherheit.

Kontrolle

Sie wissen jetzt, dass Sie für die Übertragung von Dateien auf einen Webserver ein FTP- oder ein SSL-Programm benötigen. Sie wissen auch, dass Sie

im Netz nicht nach FTP-Programm oder SSL-Programm suchen, weil Sie wissen, dass es für solche Programme eine andere Bezeichnung als das Wort Programm gibt.

Wie Sie ein solches Programm bedienen müssen ist nicht Teil dieser Veranstaltung, da auch hier erneut weitergehende Kenntnisse bei der Administration von Servern nötig sind, die mit der Entwicklung einer Webanwendung bzw. den Grundlagen der Programmierung nichts zu tun haben.

Stichwortverzeichnis