Web Application Development

Organisatorisches, Scheinkriterien, Einführung

Martina Freundorfer

Kontakt

Dipl.-Inf. (FH) Martina Freundorfer

Nebenberufliche Lehrbeauftragte HTW Berlin

Lehrerin für Web Development am Digital Career Institute

- Studiengang: Angewandte Informatik
  - Web Application Development
- Email: freundo@htw-berlin.de
- Oder einfach im Slack-Channel htw-berlin-hq.slack.com :)

Zu mir

- Seit 2020 an der HTW Berlin
- Seit 2019 unterrichte ich Web Development am Digital Career Institute gGmbH auf Englisch, Schwerpunkte UI Basics, Programming Basics, Browser
- 2018: Senior Web Engineer in einer Werbeagentur
- 2015 – 2017: Web Developer bei Bosch Software Innovations
- 2010 – 2015: Web Developer bei eBay Inc./mobile.de GmbH
- Umzug nach Berlin aus privaten Gründen
- 2007 – 2010: Software-Entwicklerin bei GMX / 1&1 Internet AG
- 2003 - 2007: Diplom-Informatikstudium an der FH München
- Ursprünglich aus München, Bayern

Und Sie?

- Wie heißen Sie?
- Wo sind Sie aufgewachsen, oder wo kommen Sie her?
- Welches Gebiet der Informatik interessiert Sie, auch bzgl. zukünftigen Jobs oder Masterstudium?
  - Mobile Anwendungen
  - Webanwendungen, Frontend oder Backend
  - Verteilte Systeme
  - Data Science, KI
  - Computer Graphics, Multimedia
  - Game Development
  - Gesundheitsinformatik
  - ...

Heute

- Organisatorisches, Scheinkriterien, etc.
- Was werden wir in Web Application Development in SoSe 2020 machen?
- Übung nach der Vorlesung heute:
  - Die Webapplikation, die wir in WAD erstellen werden, vorstellen
  - Dazu ein erstes npm-Projekt erstellen
  - Live Server installieren

Etikette

- Bitte keine Handys!
  - Handy lautlos
  - Dem Zwang, auf’s Handy gucken zu müssen, bitte widerstehen!
  - Deshalb am besten gleich wegpacken oder Flugmodus
- Bitte pünktlich sein!
- Mikros immer muten! Es spricht immer nur eine Person!
- Trotz Home-Office bitte passend anziehen ;-)
- Bitte lesen:
  - **http://wiw-fernstudium.htw-berlin.de/files/Stg/WIW/Verhaltenskodex/30_Tipps_fuer_WIW-Studierende.pdf**
    (keine 30, eher 8 Tipps, nicht nur für WIW-Studis, hat auch nichts mit Fernstudium zu tun)
  - htw-berlin.de: Suche nach “Verhaltensrichtlinien”
  - **http://www.mz-web.de/wirtschaft/finanzen/karriere/anrede--sprechstunde-sieben-knigge-regeln-fuer-die-uni-1447036**
  - **https://www.scientificamerican.com/article/a-learning-secret-don-t-take-notes-with-a-laptop/**

Vorlesungs- und Übungszeiten

- Zug 2

  - Vorlesung: Fr, 09:45 – 11:15 Uhr im Zoom / Raum C 444
    https://zoom.us/j/419950259

  - Übung Gr. 1: Fr, 12:15 – 13:45 Uhr im Zoom / Labor C 635 (wöchentlich)
    https://zoom.us/j/472098033

  - Übung Gr. 2: Fr, 14:00 – 15:30 Uhr im Zoom / Labor C 635 (wöchentlich)
    https://zoom.us/j/107699422

  - **Übungen für beide Gruppen finden wöchentlich statt!**
  - **Immer die gleichen Zoom URLs für alle Veranstaltungen!**

Github, Folien, Quellenangaben

- Foliensätze für die Vorlesungen in Github
- Aufgabenblätter für die Übungen in Github

- Literaturangaben in den Folien dienen als:
  1. Referenzangaben für Inhalte der Vorlesung
  2. Artikel, Links oder Bücher zur Vertiefung des Stoffes
  3. Interessante und relevante Artikel
  4. Hilfe für Übungsaufgaben

Vorlesungen

- Fangen pünktlich an, deswegen: Bitte pünktlich kommen!
- Wiederholung des Stoffes der letzten Woche: alle zusammen oder in Teams/Paaren oder Quiz
- Danach neuer Stoff, Theorie und viel Beispielcode
- Beispielcode wird vor der Vorlesung in Github verfügbar sein
- Beispielcode muss nicht vollständig sein, meist “stubs”, welche im Unterricht vervollständigt werden, deswegen:

  Laptop mitbringen und mitmachen

Vorlesungen

- Folien und Code werden nach der Vorlesung in Github verfügbar sein
- **Bitte kommen Sie vorbereitet zur Vorlesung!**
  - **letzte Vorlesung angucken, ggf. Fragen stellen**
  - **Fragen zu den Belegen können auch besprochen werden**
  - **Neue Vorlesung herunterladen**
  - **Den neuen Beispiel-Code herunterladen und in Ihr IDE importieren!**

Übungen

- Aufgabenblätter für die Übungen in Github
- Übungen sind Programmierungsaufgaben zur praktischen Vertiefung des Lernstoffs
- Alle Aufgaben sind Belegaufgaben
- Belegaufgaben werden abgegeben, präsentiert, bewertet und sind Teil Ihrer Gesamtnote
- Alle Aufgaben werden in Paaren bearbeitet
- Aufgaben können in den Übungen nicht fertiggestellt werden, aber nutzen Sie die Übungszeit mit Ihrem/r Partner/in zusammen
- Code für alle Aufgaben muss ins Github
- Zu den Belegaufgaben später mehr

Scheinkriterien

- Gesamtnote setzt sich aus den Punkten für Belegaufgaben zusammen:

  ∑ PkteBelegaufgaben / 3 = Gesamtpunktzahl

- Summe der erreichbaren Punkte bei Belegaufgaben = 100

Scheinkriterien

- Beispiel a):
  - ∑ PkteBelegaufgaben / 3 = 80 von 100 (80%)
  - → Bestanden!
- Beispiel b):
  - ∑ PkteBelegaufgaben / 3 = 35 von 100 (35%)
  - → Nicht bestanden!
- Beispiel c):
  - ∑ PkteBelegaufgaben / 3= 90 von 100 (90%)
  - Bestanden!

Scheinkriterien: Belegaufgaben

- In Form von User Stories
- Fertigstellungstermin ist ein Übungstermin, meist 2-3 Wochen Zeit
- Werden in den Übungen zusammen besprochen und ggf. legen wir zusammen fest, welche Anforderungen zu implementieren sind, diese Anforderungen gelten dann für diese Übungsgruppe
- **Sind in Paaren zu bearbeiten (Keine Ausnahmen!)**
- Beide wirken an jeder Belegaufgabe mit, jede/r Partner/in zu 50%
- Bearbeitung der Belegaufgaben darf nicht aufgeteilt werden, also NICHT: Ich mache Beleg 1 & 2, du machst 3
- Programmierlösungen für die Belegaufgaben müssen auf dem eigenen Rechner präsentiert werden
- Entsprechender Code ist termingerecht im Github eingecheckt

Scheinkriterien: Belegaufgaben

- Code muss zum Fertigstellungstermin im github sein: letzter Commit vor Übungsbeginn am Tag der Präsentation
- Das Team muss die Belegaufgabe vorstellen:
  - Der Code ist auf dem Rechner ausgecheckt
  - Build läuft durch
  - Code muss auf dem Rechner laufen, in Kommandozeile oder im Live-Server
  - Beide Mitglieder müssen das Programm, den Code und ggf. Tests erklären können
- Wenn zur Präsentation das Build nicht durchläuft oder das Programm nicht läuft, oder der Live-Server ist nicht installiert, dann gibt es 0 Punkte für diesen Beleg!
- Git commits:
  - Ca. 50% des Codes vom/n Partner/in in jedem Beleg (ca. 50/50)
  - Beteiligt sich ein/e Partner/in gar nicht (absolut keine Commits), dann bekommt diese Person keine Punkte für den Beleg
  - Beteiligt sich eine Person sehr wenig an einem Projekt, dann gibt es eine Unterredung zusammen mit dem/r anderen Partner/in und es wird entschieden, wie weiterverfahren wird

Scheinkriterien: Belegaufgaben

- Was ist, wenn beide am Präsentationstermin krank sind?
  - Beide ärztlichen Atteste vorgelegen, und neuen Termin absprechen
  - Kein oder nur ein Attest: Beleg zählt als nicht abgegeben
- Was ist, wenn eine/einer am Präsentationstermin nicht da ist?
  - **Sie dürfen bei einer Belegabgabe fehlen.** Es werden Ihnen trotzdem alle Punkte Ihres Teams gutgeschrieben.
  - Fehlen Sie mehr als einmal, erhalten Sie insgesamt nur die Punkte, bei denen Sie anwesend waren.
- Voraussichtlich: 100 Punkte ist die Gesamtpunktzahl für Belege, insgesamt 3 Belege, die aufeinander aufbauen und am Ende eine funktionierende Webapplikation darstellen

Warum Paararbeit, Demos und git?

- weil es so in der Praxis gemacht wird
- Software-Entwicklung ist Teamarbeit
- Agile software development:
  - Pair Programming
  - Code Reviews
  - Demos
- Pair Programming: Vier Augen sehen mehr als zwei
- Code Reviews: Noch mehr Augen sehen noch mehr :-), Bugs finden, Coding Standards eingehalten?, Tipps für “Profi”-Code
- Demos:

1. Review einer fertigen User Story durch den Product Owner
2. Präsentation am Sprintende mit allen beteiligten Teams

Aufgaben bis übernächste Woche:

Aufgabe 1:

- Teampartnersuche: Person suchen, mit der Sie in diesem Semester in WAD zusammenarbeiten werden
- Tragen Sie sich als Team in die Liste in den Übungen ein, spätestens bis übernächste Woche

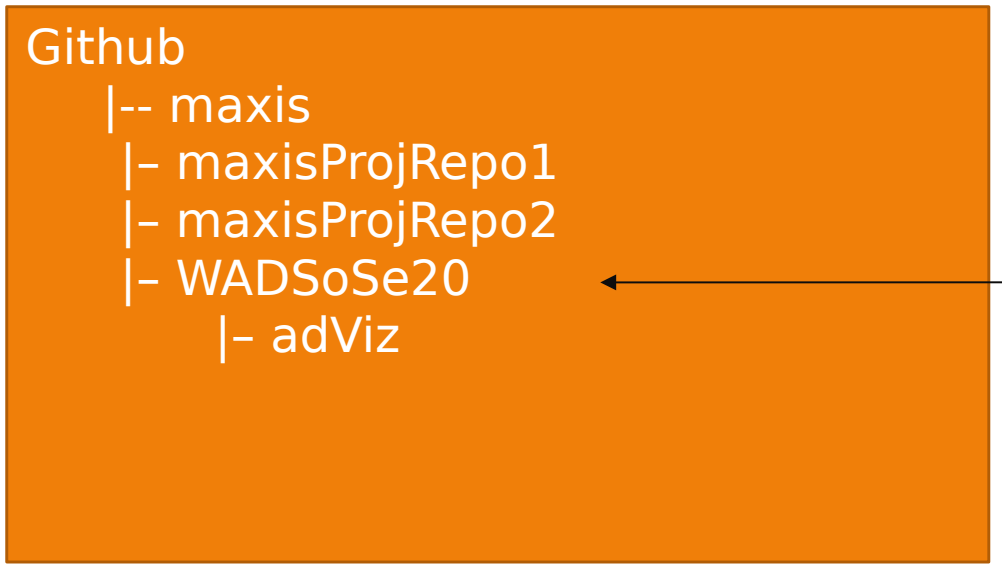
Wir werden github für die git-Repositories nutzen

Aufgabe 2:

- Falls Sie noch keinen github-account haben, dann bitte einen anlegen.
- Als Studi, mit Ihrer “htw-berlin.de”-Emailadresse, können Sie sich private Repositories anlegen, ohne dafür Geld zahlen zu müssen
- Ein Teammitglied richtet sich bitte ein privates Repository in github mit Namen “WADSoSe20” ein
- Bitte mich, martl, als Collaborator Ihrem Repo hinzufügen

github

- Ein Teammitglied richtet sich bitte ein privates Repository in github mit dem Namen “WADSoSe20” ein
- Beispiel:
  - Max Mustermann und Maxime Musterfrau wollen zusammenarbeiten
  - Maxime’s github-login ist “maxis”
  - Maxime legt das Repo ‘WADSoSe20’ an, und fügt Max und ‘martl’ als Collaborator dem git-Repo ‘WADSoSe20’ hinzu
  - Damit hat maxis folgendes Verzeichnis in github:



Das ist das private Repo, welches maxis angelegt hat und zu dem Teampartner “Max” und ich, “martl”, als Contributers eingeladen sind.

Agile SW-Entwicklung: Scrum Board

ToDo	In Progress	Code Review	PO Review	Done
User Story 1	User Story 1	User Story 1	User Story 1	User Story 1
	• Team erarbeitet Subtasks • Im git wird ein Branch names “UserStory_1” angelegt • Teammitgliede r committen Code in diesen Branch	• Branch “UserStory_1” soll in den “master”-Branch gemergt werden • Dazu wird ein “Pull Request” in github angelegt • Teammitglieder können Code kommentieren	• Fertige User Story wird dem PO präsentiert • Wenn alles ok, dann Mergen des Branches in den “master”-Branch	

SW-Entwicklung in WAD

ToDo	In Progress	Review- Part 1 (Belegabgabe)	Review – Part 2	Done
Beleg 1	Beleg 1	Beleg 1	Beleg 1	Beleg 1
	• Team erarbeitet Teilaufgaben • Teammitglieder committen Code in den master-Branch	• Belegabgabe	• martl guckt sich Code in github an	

Paararbeit

- In den Übungen:
  - ein Rechner für zwei Personen oder
  - zwei nebeneinander stehende Rechner
  - Laptop ist auch ok (Belegabgabe auf Laborrechnern: nicht vergessen!!)
- Remote pair programming mit folgenden Werkzeugen:
  - Zoom (maximal 40 Minuten ohne Pro-Account), Slack (Video und Audio möglich), Google Hangouts, Jitsi etc.pp.
  - VSCode Live-Share Plugin um zusammen zeitgleich am gleichen Code arbeiten zu können (inklusive Browser Sharing)
  - GitHub ;-)

Anwesenheit

- Präsenz- und Vollzeitstudium
- Falls Sie während einer Vorlesung oder Übung abwesend sind, dann sind Sie für den versäumten Stoff bzw. Informationen verantwortlich. Das gilt besonders für Übungen, in denen die Anforderungen für die Belege besprochen werden. Belegaufgaben werde ich rechtzeitig in Github hochladen. Also, “Das wusste ich nicht, da war ich nicht da.” zählt nicht. Informieren Sie sich bitte bei Ihren Mitstudierenden.
- Nutzen Sie die Übungen, um als Team an den Projekten zu arbeiten. Sie können die Tools/das Labor natürlich auch ausserhalb der Übungszeiten nutzen.
- Belegabgabetermine sind einzuhalten! Wie schon besprochen, kann ein Partner zu einem Abgabetermin einmal abwesend sein.

Fragen?

- Haben Sie Fragen zu dem Ablauf der Vorlesung bzw. Übungen?
- Oder zu den Scheinkriterien, Belegaufgaben?

Was werden wir in WAD in SoSe2020 machen?

#	KW	Datum	Vorlesung	**Das Labor nach der Vorlesung findet für beide Gruppen wöchentlich statt.**
1	14	3.4.	Einführung, Scheinkriterien	npm im Selbststudium, Anlegen eines Webprojektes, Installation von Live Server
2	16	17.4.	Client-Server, Web Apps, URI, HTTP	Aufgabe von Beleg 1: AdViz (nur HTML und CSS)
3	17	24.4.	HTML, CSS	Prototyp AdViz
4	20	15.5.	CSS, JavaScript allgemein	Abgabe: Prototyp AdViz
5	22	29.5.	JavaScript: DOM, JSON, AJAX	Aufgabe von Beleg 2: Adviz mit JS
6	23	5.6.	React Framework	AdViz mit JS
7	24	12.6.	React Framework	AdViz mit JS
8	25	19.6.	React Framework	Abgabe: AdViz mit JS
9	26	26.6.	React Framework / NodeJS	Aufgabe von Beleg 3: AdViz mit Backend
10	27	3.7.	NodeJS	AdViz mit Backend
11	28	10.7.	NodeJS	AdViz mit Backend

Wie alt ist das Internet?

- SurveyMonkey-Umfrage: “Alter des Internets”
- https://www.surveymonkey.de/r/FRCCNDP

Was ist das Internet?

- Internet, eigentlich “internetwork”, ist ein weltweiter Verbund von Rechner-netzwerken
- Da Rechnernetzwerke weltweit miteinander verbunden sind, können sich einzelne Rechner mit anderen Rechnern verbinden, und unter anderem Internetdienste wie
  - das World Wide Web
  - E-Mail
  - Telnet
  - FTP

  nutzen
- Für den Datenaustausch zwischen den Rechnern werden Internetprotokolle verwendet.
- Übertragung von Daten im Internet unabhängig von ihrem Inhalt, dem Absender und dem Empfänger wird als Netzneutralität bezeichnet.

Was ist das Internet?

- 1969: Das Internet ging aus Arpanet hervor, welches im Jahr 1969 entstand.
- Arpanet war ein Projekt der Advanced Research Project Agency (ARPA), ARPA gehörte dem Department of Defense (DoD) an
- Arpanet sollte Universitäten und Forschungsinstitute vernetzen und die knappen Kapazitäten der teueren Großrechner besser nutzen
- Kommunikationsprotokolle waren ungeignet für heterogene Umgebungen
- 1981: Entwicklung der TCP/IP-Netzwerkprotokolle in den 1980er
- 1984: Entwicklung des Domain Name System (DNS) ermöglichte die Adressierbarkeit von Rechnern mit Namen, die **man (Mensch)** sich merken konnte, z.B.: cern.ch
- 12.03.1989: Tim Berners-Lee (CERN) stellt die Idee des World Wide Web vor, ursprüngliches Ziel: Forschungsergebnisse mit Kollegen einfacher austauschen

World Wide Web

- Tim Berners-Lee: “Das World Wide Web ist eine großräumige Hypermedia-Initiative zur Informationsbeschaffung mit dem Ziel, den allgemeinen Zugang zu einer großen Sammlung von Dokumenten zu erlauben.”
- Wikipedia: Das **World Wide Web** ist ein über das Internet abrufbares System von elektronischen Hypertext-Dokumenten, den Webseiten. Sie sind durch Hyperlinks untereinander verknüpft und werden im Internet über die Protokolle HTTP oder HTTPS übertragen. Die Webseiten enthalten meist Texte, oft mit Bildern und grafischen Elementen illustriert. Häufig sind auch Videos, Tondokumente und Musikstücke eingebettet.
- Umgangssprachlich wird WWW mit dem Internet gleichgesetzt, aber es ist jünger und eine von mehreren Nutzungen des Internets
- 1991: erste Webseite - http://info.cern.ch/hypertext/WWW/TheProject.html
- 1993: erster grafikfähige Browser “Mosaic”, der die Darstellung von Inhalten des WWW ermöglichte, kostenlos

World Wide Web

- **Basiert auf drei Kernstandards:**
  - **HTTP** = Hypertext Transfer Protocol als Protokoll, mit dem der Browser Informationen vom Webserver anfordern kann
  - **HTML** als Auszeichnungssprache: Gliederung der Information, Verknüpfung von Dokumenten durch Hyperlinks
  - **URLs** als eindeutige Bezeichnung einer Resource, die in Hyperlinks verwendet wird.
- Später kommen dazu:
  - Cascading Style Sheets (CSS) legen das Aussehen der Elemente einer Webseite fest
  - HTTPS = Hypertext Transfer Protocol Secure, eine Weiterentwicklung von HTTP, dient dem verschlüsselten Datentransfer
  - Document Object Model (DOM) als Programmierschnittstelle für externe Programme oder Skriptsprachen von Webbrowsern.
  - JavaScript - eine Skriptsprache mit Anweisungen für den Browser

World Wide Web

- Erste persönliche Webseiten, mittlerweile über 19 Jahre alt:

  http://web.archive.org/web/20010929233155/http://martl.de.vu/

World Wide Web

- Dynamische Webseiten und Webanwendungen
- Können durch den Webserver oder/und im Browser erzeugt werden:
  - Server-seitig: Erzeugung des Inhalts der Webseite durch
    - CGI-Skripte (PHP, Perl)
    - kompilierte Anwendung (JSP, Servlets, .NET, Spring MVC)
  - Client-seitig: der Inhalt der Webseite wird mittels JavaScript erzeugt oder geändert
  - Gemischte Ausführung: AJAX (Asynchronous JavaScript and XML) – Browser sendet mittels JavaScript einen Request an den Webserver, Webserver schickt gewünschte Daten zurück, JavaScript erneuert Teile der HTML-Struktur mit diesen Daten
- Webapplikationen, an denen ich mitgewirkt habe: http://firstsearch.oclc.org, https://www.mapquest.com
- Mittlerweile sind Webanwendungen so interaktiv und allgegenwärtig, dass sie traditionelle Desktopapplikationen ersetzen: Mailclients, Google's Hangouts, Calendar, Drive, usw.. Solche Webapps werden auch als Rich Internet Applications = RIAs bezeichnet

Wie sieht es mit Ihrer Erfahrung in der Webentwicklung aus?

- SurveyMonkey-Umfrage “Web Tech”:

https://www.surveymonkey.de/r/FX7Y8KZ

Frameworks

- Welche Frameworks kennen Sie?
- Mit welchen Frameworks haben Sie schon gearbeitet?

Frameworks

- Was genau ist ein Framework?

Frameworks

Ralph E. Johnson, Brian Foote (1988):

- Ein Framework ist eine semi-vollständige Applikation.
- Es stellt für Applikationen eine wiederverwendbare, gemeinsame Struktur zur Verfügung.
- Die Entwickler bauen das Framework in ihre eigene Applikation ein, und erweitern es derart, dass es ihren spezifischen Anforderungen entspricht.
- Frameworks unterscheiden sich von Toolkits (oder Bibliotheken) dahingehend, dass sie eine zusammenhängende Struktur zur Verfügung stellen, anstatt einer einfachen Menge von Hilfsklassen.

Frameworks

- Rahmen, innerhalb dessen der SW-Engineer eine Anwendung erstellt
- Struktur der individuellen Anwendung wird durch Framework beeinflusst
- Entwickler implementiert bzw. erweitert Schnittstellen oder Klassen des Frameworks, und registriert diese
- Framework steuert und ruft diese Implementierung auf -> Hollywood Principle: "Don't call us, we'll call you."
- Wiederverwendbare, gemeinsame Struktur
- Meist für ein bestimmtes Anwendungsgebiet
- Sind nicht nur Bibliotheken

Frameworks

- Web Frameworks: **React Framework**, Spring MVC, Java Server Faces, etc.
- Testing Frameworks: JUnit, TestNG, Selenium
- Logging Frameworks: Apache Log4j, LogBack
- JSON Processing: Jackson (JSON = JavaScript Object Notation ist ein Datenformat)

The End

- Fragen?
- Referenzen und Literaturhinweise:
  - Dane Cameron: “A Software Engineer Learns HTML5, JavaScript & jQuery”, Cisdal Publishings, 2015
  - https://webfoundation.org/2018/03/web-birthday-29/
  - https://www.reporter-ohne-grenzen.de/pressemitteilungen/meldung/uncensored-playlist-mit-pop-songs-die-zensur-umgehen/