

- **Organisatorisches**
 - Ablauf der Übung
 - Themen
 - Abgabemodalitäten
 - Gruppenzuordnung
- **Aufgabenstellung**
 - Hinweise zur Umsetzung
- **Einstieg ins Thema**
 - JavaScript
 - AJAX
 - Beispiele
- **Links**

■ Terminplan

- **05.01.12 Übungseinführung 2B (Pflicht)**
- 12.01.12 Konsultation (Teilnahme freiwillig)
- 19.01.12 Konsultation (Teilnahme freiwillig)
- 26.01.12 Konsultation (Teilnahme freiwillig)
- **30.01.12 Abgabe der Ergebnisse (bis spätestens 13:00 Uhr)**
- ...
- 01.02.12 Abschlussveranstaltung und Präsentation (Pflicht)

■ Details siehe [Übungskalender](#)

■ Themenkomplex 1

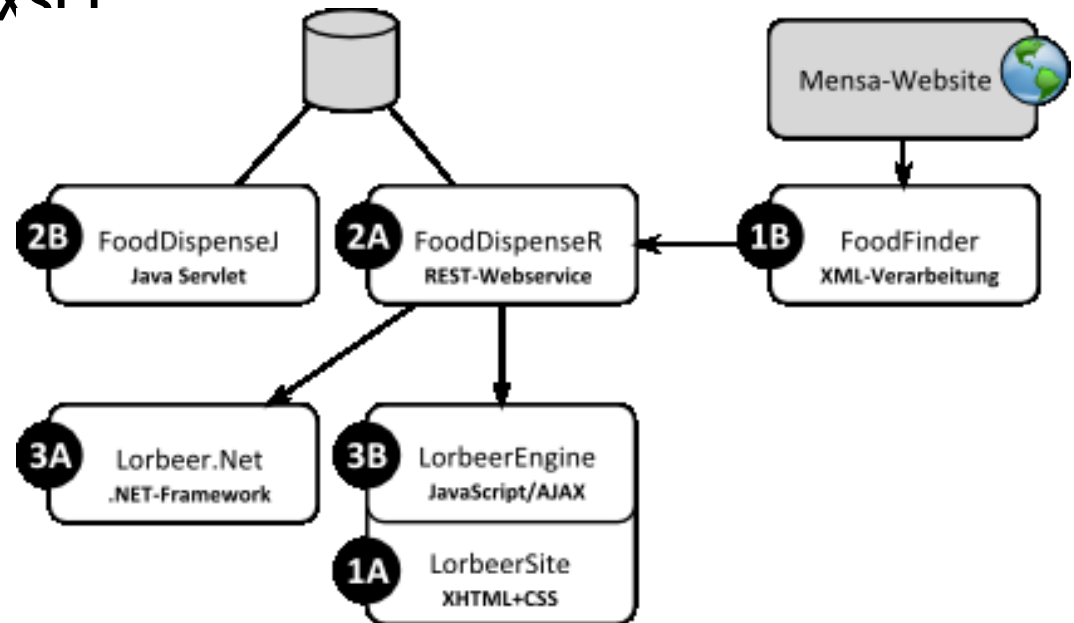
- 1A) XHTML und CSS
- 1B) XML-Verarbeitung und XSL T

■ Themenkomplex 2

- 2A) RESTful Webservice
- 2B) Java Servlets

■ Themenkomplex 3

- 3A) .NET/Silverlight
- 3B) JavaScript und AJAX



■ Informationen zu den Übungen

http://www.mmt.inf.tu-dresden.de/Lehre/Wintersemester_11_12/WME/Uebung/index.xhtml

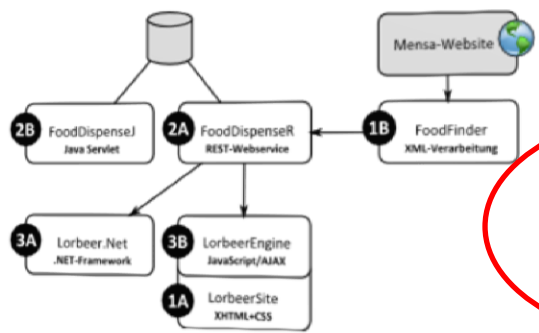
Übung zur Vorlesung
Web- und Multimedia-Engineering

[Dipl.-Medieninf. Matthias Niederhausen](#)
 siehe unten; zuerst in der Woche vom 17.10.2011
 INF E069
 0/2/0/0 SWS

Kurzbeschreibung

Zur Ergänzung und zum Verständnis des Vorlesungsstoffes werden drei Themenkomplexe mit jeweils zwei Aufgabenstellungen angeboten, die sich alle mit speziellen Teilproblemen einer übergeordneten Beispielanwendung beschäftigen. Jeder Student bearbeitet je eine Aufgabe aus den Themenkomplexen 1, 2 und 3 in einer Übungsgruppe (2 Studenten). Bei jedem Themenkomplex hat man die Wahl zwischen Thema A oder B. Man kann A und B bei jedem Themenkomplex neu wählen, also 1A, 2B und 3A sind durchaus möglich. Für die Bearbeitung einer Aufgabe haben die Gruppen etwa vier Wochen Zeit. Danach werden die Lösungen durch die Gruppen präsentiert (ca. fünfminütiger Vortrag). Wenn eine Gruppe gerne eine Projektplattform nutzen möchte, ist z.B. [assembla](#) zu empfehlen.

Außer bei den Einführungsveranstaltungen wird die Übung in selbstständiger Arbeit abgehalten; der Tutor steht bei eventuellen Fragen zur Seite. Teilweise werden wiederkehrende Probleme auch für alle Übungsteilnehmer erklärt. Die Einschreibung erfolgt für einen der vier angebotenen Termine, da die Räume eben nur begrenzt Platz bieten. Im Normalfall erfolgt die Einschreibung sich also für einen Termin, der auch mit dem Wahlthema übereinstimmt. Im Ausnahmefall ist es auch möglich, zu den anderen Terminen die Übungen besuchen. Dort wird man aber primär mit den Inhalten des anderen Themas konfrontiert, der Tutor wird natürlich trotzdem versuchen zu helfen.



Inhalt

- [Kurzbeschreibung](#)
- [Aktuelles](#)
- [Übungszeiten und Tutoren](#)
- [Terminplan](#)
- [Einschreibung](#)
- [Materialien](#)
- [Abgabe](#)
- [Bewertung](#)

Nachgefragt

Noch keine Fragen vorhanden...

[Alle Fragen ansehen](#)

[Frage stellen](#)

Forum für Nachfragen

Nachgefragt
 Noch keine Fragen vorhanden...
[Alle Fragen ansehen](#)
[Frage stellen](#)

- via Upload auf **sFTP** Server:
 - Servername: **serv9.inf.tu-dresden.de**
 - Port **22**
 - Verzeichnis: **/zbv/WME/**
 - Authentifizierung mit **ZIH-Login**

- Abgabe der Lösung
 - Verzeichnis anlegen
 - Ergebnisse_3_B/<Gruppe>_<Matrikel1>_<Matrikel2>
 - Dateien hochladen
 - keine Umlaute in Dateinamen!
 - Zugriffsrechte nicht verändern!




■ Lösung der Aufgabe






- **zwei Studenten** bilden ein Team
 - auf der Einschreibliste eintragen!
- Durchschleifen wird nicht toleriert
- Abgabetermin ist verbindlich (**30.01.2012 - bis 13Uhr**)
 - Abgabeverzeichnisse werden nach Ablauf der Bearbeitungszeit gesperrt
- jede Gruppe hat eine **eigenständige** Lösung abzugeben
- ausgewählte Ergebnisse werden am letzten Übungstermin (**01.02.2012**) von den Gruppen präsentiert

■ Aufgabe 3B: Erweiterung eines XHTML-Mockups mit AJAX-Funktionalität

- Das Portal Lorbeerblatt soll Studenten eine Plattform bieten, auf der sie sich über das aktuelle Mensaessen austauschen können. Die Benutzer sollen Kommentare, Bewertungen und eigene Bilder der verschiedenen Speisen einstellen können. Ihre Aufgabe ist es, das Mockup auf der Übungsseite so mit JavaScript zu erweitern, dass die angezeigten Daten tatsächlich vom Server geladen werden und Nutzer eigene Inhalte hinzufügen können.

Mensa-Speiseplan vom Freitag, den 14. Oktober 2011

Neue Mensa	Infos	Preise
Schinkenmakkaroni mit fruchtiger Tomatensoße und geriebenem Käse (auch ohne Schinken möglich)		1,66 € / 3,16 €
Schweinesteak mit Kräuterbutter dazu Pommes frites oder Risoleekartoffeln und Salat		ausverkauft
Wok: Pizza mit Zucchini und Hirtenkäse ---AUCH als halbe Pizza: 1,62€/ 2,62€-- -		2,43 € / 3,43 €
Weitere Angebote und Informationen anzeigen ▼		

Alte Mensa	Infos	Preise
Germknödel mit Heidelbeerfüllung, dazu Vanillesoße		1,70 € / 3,20 €
Hausgemachte Schweinsröllchen mit Schinkenfüllung, dazu Möhren-Fenchelgemüse und Petersilienkartoffeln	  	2,20 € / 3,70 €
Gebratene Spätzle mit Gemüsestreifen und Käsesoße, dazu Salat	 	1,70 € / 3,20 €
Weitere Angebote und Informationen anzeigen ▼		

- Entwicklung der **LorbeerEngine** für Lorbeerblatt
 - Clientseitige Verarbeitung der Daten vom Webservice durch JavaScript und AJAX
 - Funktionen:
 - Darstellung der nach Nutzerkriterien gefilterten und sortierten Liste von Speisen des Tages
 - Anzeige eines einzelnen Gerichts mit Bewertungen, Nutzerbildern und Kommentaren
 - Upload eines Fotos zu einem Gericht
 - Bewertung eines Gerichts
 - Kommentierung eines Gerichts

■ Randbedingungen

- **Lauffähige Anwendung:** Die abgegebene Anwendung soll ohne weitere Konfigurationen unmittelbar lauffähig sein!
- XHTML-Mockup muss **reale und korrekte** Daten der Mensawebsite beinhalten
 - <http://www.studentenwerk-dresden.de/mensen/speiseplan/>
- **Lizenzrechte** bei verwendeten Fremdinhalten beachten!

■ Abgabe

- gepackte Quellen (z.B. als gepacktes Verzeichnis)
- kurze Ausführungsbeschreibung

■ Material

■ FoodDispenser.war

- REST-Webservice, der über den FoodFinder alle Mensadaten bereitstellt
- muss in einem lokalen Tomcat „deployed“ werden
- Beschreibung des Webservices (WADL) unter <http://localhost:8080/FoodDispenser/application.wadl>

■ LorbeerSite.zip

- enthält ein XHTML-Mockup, welches in Übung 1A entwickelt wurde
- muss entpackt und nach dem Start des Tomcats (er erzeugt aus der FoodDispenser.war einen neuen Ordner unter */webapps*) unter *tomcat/webapps/FoodDispenser/lorbeersite* gespeichert werden
- die Mockup-Seite ist dann unter <http://localhost:8080/FoodDispenser/lorbeersite/index.html> erreichbar

- Editoren und Entwicklungsumgebungen
 - Notepad++
 - Texteditor mit Syntax-Highlighting
 - klein und schnell
 - zur Ansicht und Bearbeitung der JavaScript- und XHTML-Dateien
 - Firebug (Mozilla Firefox)
 - Sammlung von Entwicklungstools
 - Editieren, Debuggen, Monitoren von CSS, HTML und JavaScript
 - Open HttpRequester / Open Poster (Mozilla Firefox)
 - Manipulation von HTTP-Anfragen (GET, PUT, POST, DELETE)
 - Darstellung der Response-Daten
 - History-Funktion für Transaktionen

■ Tomcat + Webservice

- Tomcat 7.0 runterladen <http://tomcat.apache.org/download-70.cgi> und entpacken
- FoodDispenser.war in den Ordner {tomcat}/webapps kopieren und Tomcat mit startup.bat in dem bin-Ordner starten
- nach dem erfolgreichen Start, Tomcat mit shutdown.bat in dem bin-Ordner runterfahren und die FoodDispenser.war in webapps löschen

■ XHTML-Mockup

- LorbeerSite.zip entpacken und den Inhalt in einen neuen Ordner „lorbeersite“ unter {tomcat}/webapps/FoodDispenser/lorbeersite kopieren
- Tomcat starten und folgende Seite aufrufen:
<http://localhost:8080/FoodDispenser/lorbeersite/index.html>

■ Warum ist das so kompliziert?

→ „Same-Origin-Policy“

- Sicherheitskonzept, das es nur erlaubt auf Objekte derselben Quelle zuzugreifen
- Zugriff vom Filesystem per JavaScript auf <http://localhost:8080> ist nicht erlaubt

■ Wie kann das Problem umgangen werden?

- Anfrage über eine proxy.php an den Webservice stellen (php ist eine serverseitige Skriptsprache)
- Verwendung von JSONP (serverseitige Implementierung notwendig)
- **Laden der Seiten in den Tomcat (aktuelle Umsetzung)**

■ RESTful Webservice:

- Aufruf der WADL zur Ableitung der Webservicefunktionalitäten:

<http://localhost:8080/FoodDispenser/application.wadl>

```
<ns2:application>
  <ns2:doc jersey:generatedBy="Jersey: 1.9.1 09/14/2011 02:36 PM"/>
+ <ns2:doc xml:lang="de" title="FoodDispenser API (WADL)"></ns2:doc>
+ <ns2:grammars></ns2:grammars>
- <ns2:resources base="http://hyperadapt.net/FoodDispenser/">
  + <ns2:resource path="/mensas"></ns2:resource>
  + <ns2:resource path="/dishes"></ns2:resource>
  + <ns2:resource path="/users"></ns2:resource>
  + <ns2:resource path="/photos"></ns2:resource>
  + <ns2:resource path="/comments"></ns2:resource>
  + <ns2:resource path="/ratings"></ns2:resource>
  </ns2:resources>
</ns2:application>
```

■ JQuery – GET Anfragen:

■ Synchron:

```
$.get(  
    'http://localhost:8080/FoodDispenser/mensas',  
    function(data) {  
        var xml = data;  
        var mensas = xml.getElementsByTagName("mensas");  
    }  
);
```

■ Asynchron:

```
$.ajax({  
    url: 'http://localhost:8080/FoodDispenser/mensas',  
    data: '',  
    dataType: 'xml',  
    success:  
        function(data) { ... }  
});
```

- JQuery – POST Anfragen:

- Mensa hinzufügen:

```
$.ajax({
  type: "POST"
  url: 'http://localhost:8080/FoodDispenser/mensas',
  data: '<mensa><name>TestMensa</name></mensa>',
  dataType: 'xml',
  success:
    function(result) {
      alert(result);
    }
});
```

■ JQuery – weitere wichtige Attribute:

```
$.ajax({
  type: "GET"
  url: 'http://localhost:8080/FoodDispenser/mensas',
  data: '',
  dataType: 'json',
  contentType: "application/json",
  header: '{Accept: application/json}',
  success:
    function(data) {
      var xml = data;
      var mensas = xml.getElementsByTagName("mensas");
    }
});
```

- mehr dazu unter: <http://api.jquery.com/>

■ Beispiel - Element hinzufügen:

■ `<div id="mensas"></div>`

```
// Mensaknoten aus dem XHTML-Dokument holen
var divMensaElem = document.getElementById('mensas');

// neues div-Element erzeugen und dem Mensaknoten anhängen
var newDiv = document.createElement('div');
var newDivElem = divMensaElem.appendChild(newDiv);

// Attribute setzen
newDivElem.id = "123";
newDivElem.className = "mensa";
```

■ hilfreiche Dokumentationen:

<http://www.peterkropff.de/site/javascript/javascript.htm>

<http://de.selfhtml.org/javascript/index.htm>

- **Tutorien & Co. (JavaScript, AJAX, JQuery)**
 - <http://www.peterkropff.de/site/javascript/javascript.htm>
 - <http://de.selfhtml.org/javascript/index.htm>
 - <http://www.w3schools.com/js/>
 - <http://api.jquery.com/>
 - <http://www.google.de>

- **Kontakt**
 - Martin.Lehmann4@mailbox.tu-dresden.de