Hochschule Darmstadt
Fachbereich Informatik

Entwicklung webbasierter Anwendungen

Praktikumsaufgaben

Semesterthema "Webbasierter Pizzaservice"

- Im Lauf des Semesters soll eine integrierte webbasierte Anwendung zur Unterstützung eines Pizzaservices entwickelt werden
  - ⇨ nicht die Funktionalität steht im Vordergrund, sondern die Integration der verschiedenen Techniken und die Methodik der Vorgehensweise
    - Clientseitig
      - Webseiten mit HTML und CSS, Formularen und JavaScript
    - Serverseitig
      - Apache Webserver
      - dynamische Seiten mit PHP, Anbindung einer MySQL-Datenbank
- Schwerpunkt auf professionelle Webentwicklung
  - ⇨ mit Standardkonformität, Barrierefreiheit, Dokumentation, Test etc.
  - ⇨ keine Homepage-Bastelei, keine Verwendung von "Fertigteilen" !

Reine Funktionalität reicht NICHT für die Abnahme !

Semesterthema "Webbasierter Pizzaservice"

- Anforderungen des Auftraggebers
  - ⇨ Der Pizzaservice soll folgende Webseiten enthalten:
    - Bestellung
    - Bestellstatus
    - Pizzabäcker
    - Fahrer
  - ⇨ Der Pizzaservice soll mit HTML5 und CSS Level 3 dargestellt werden
  - ⇨ Als Webserver wird Apache 2 verwendet und als Datenbank MySQL
  - ⇨ Die Abnahme erfolgt unter Windows mit aktuellem Firefox und IE
    - installieren Sie dazu das Add-on "Web Developer Toolbar" in Firefox

Designskizzen "Webbasierter Pizzaservice"

Kunde (Bestellung)

Margherita	4,00 €	
Salami	4,50 €	
Prosciutto	5,50 €	
Tonno	5,00 €	

Warenkorb

Margherita
Tonno
Prosciutto

14,50 €

Meier, Hauptstr. 5

Alle Löschen | Bestellen

Auswahl Löschen

Kunde (Lieferstand)

	bestellt	im Ofen	fertig	unterwegs
Margherita	●	○	○	○
Tonno	○	●	○	○
Prosciutto	○	●	○	○

Neue Bestellung

Pizzabäcker (bestellte Pizzen)

	bestellt	im Ofen	fertig
Margherita	●	○	○
Tonno	○	●	○
Prosciutto	○	●	○
Salami	○	○	●
Prosciutto	○	○	●

Fahrer (fertige Pizzen)

Schulz, Kasinostr. 5 13,50 €
Margherita, Salami, Tonno

fertig	unterwegs	geliefert
○	●	○

Müller, Rheinstr. 11 10,00 €
Salami, Prosciutto

fertig	unterwegs	geliefert
●	○	○

Anforderungen des Auftraggebers "Webbasierter Pizzaservice"

- Pizzabestellung
  - Hier kann der Kunde seine Pizzen aus der Speisekarte auswählen und in einen Warenkorb übernehmen. Hier wird der Preis der Bestellung angezeigt und es muss eine Lieferadresse angegeben werden.
- Bestellstatus
  - Hier kann ein Kunde sehen, in welchem Zustand seine Pizzen sind (bestellt, im Ofen, fertig, unterwegs). Er sieht nur seine Bestellung – und keine Aufträge von anderen Kunden.
- Pizzabäcker
  - Hier werden die bestellten Pizzen angezeigt. Der Pizzabäcker kann den Status für jede Pizza von "bestellt" auf "im Ofen" bzw. "fertig" setzen. Übernimmt der Fahrer eine Pizza, so verschwindet sie aus der Liste.
- Fahrer
  - Hier werden Bestellungen mit den einzelnen Pizzen, Preis und Adresse angezeigt. Der Fahrer kann den Status der Lieferungen verändern. Eine Lieferung ist entweder "fertig", "unterwegs" oder "geliefert". Lieferungen tauchen erst beim Fahrer auf, wenn alle zugehörigen Pizzen fertig sind. Ausgelieferte Bestellungen verschwinden aus der Liste.

Anforderungen des Auftraggebers "Webbasierter Pizzaservice"

- Sonstiges
  - ⇨ Die Speisekarte kann um weitere Pizzen erweitert werden. Die Preisberechnungen ändern sich dann (ohne Neuprogrammierung)
  - ⇨ es werden nur gültige Bestellungen akzeptiert
  - ⇨ die Auswahl einer Pizza soll über einen Mausklick auf ein Pizzasymbol erfolgen
  - ⇨ Der Warenkorb kann mit den allgemein üblichen Funktionen bearbeitet werden
  - ⇨ Versuchen Sie das Layout so umzusetzen, wie es in der Designskizze dargestellt ist. Es sollen 4 getrennte Seiten entwickelt werden
  - ⇨ Die Seiten "Pizzabäcker" und "Fahrer" sollen sich automatisch aktualisieren
- Interne Anforderungen
  - ⇨ Es gibt zu Testzwecken eine weitere Webseite "Übersicht", welche (für einfache Tests) Links zu den 4 Webseiten beinhaltet
  - ⇨ Tests sollen so klar formuliert sein, dass sie automatisiert laufen könnten

1. Übung: Zielsetzung und Aufgabe

- Stellen Sie sicher, dass Sie die Aufgabe "Pizzaservice" genau verstanden haben und klären Sie offene Punkte frühzeitig
- Aufgabe: Anforderungsanalyse
  - ⇨ Analysieren Sie sämtliche Vorgaben (sowohl vom Auftraggeber als auch interne Vorgaben)
  - ⇨ Identifizieren Sie Inkonsistenzen oder offene Punkte und klären Sie die Fragen mit Ihrem Betreuer
  - ⇨ Erarbeiten Sie eine Liste von kurzen und informellen Anwendungsfällen
    - Dokumentieren Sie zu jeder User Story mit mindestens einem „Test“ wie Sie überprüfen wollen, dass die Funktionalität umgesetzt wurde
  - ⇨ Nach ca. 60 Minuten setzen Sie sich mit Ihrer Nachbargruppe zusammen und diskutieren die gefundenen User Stories
    - ergänzen Sie fehlende Stories und Tests

1. Übung: Hinweise

- Beispiel für eine User Story mit Test
  - ⇨ "Warenkorb füllen":
    - Der Kunde will, dass Pizzen in den Warenkorb übernommen werden, wenn er auf die gewünschten Pizzasymbole klickt damit er seine Bestellung durchführen kann. Der aktuelle Preis wird dann sofort angezeigt.
    - Test 1: Es werden durch Mausklicks 2 Tonno, 1 Salami und 3 Margherita bestellt. Die Pizzen sind im Warenkorb und kosten 26,50€
- Analysieren Sie die Layoutskizze
  - ⇨ Welche Funktionalitäten würden SIE von solchen Webseiten erwarten?
- Schreiben Sie zuerst nur Überschriften und Stichworte auf
  - ⇨ Verwenden Sie das Schema: *XXX will YYY damit ZZZ*
  - ⇨ formulieren Sie anschließend die Inhalte und die Tests
  - ⇨ Falls eine User Story zu lang / zu kompliziert wird
    - überlegen Sie was der Anwender eigentlich will – das ist meist relativ einfach
    - der Ablauf besteht dann oft aus Anwendersicht aus mehreren User Stories

2. Übung: Zielsetzung

Bäcker

	bestellt	im Ofen	fertig
Margherita	◉	○	○
Margherita	○	◉	○
Hawai	○	◉	○

- Bestellung
- Kunde
- Bäcker
- Fahrer

Übersichtsseite

Bestellung


Margherita 4,00 €


Salami 4,50 €


Hawaii 5,50 €

test

15,70 €

Alle Löschen | Auswahl Löschen | Bestellen

Fahrer

Müller, Freßgasse 11, 65000 Frankfurt

Tonno, Calzone, Margherita, Hawaii, Tonno

Preis: 13,00 €

gebacken unterwegs ausgeliefert

○ ◉ ○

Meier, Hauptstr. 5

Tonno, Tonno, Margherita

Preis: 10,50 €

gebacken unterwegs ausgeliefert

○ ◉ ○

Kunde

	bestellt	im Ofen	fertig	unterwegs
Margherita	○	○	◉	○
Salami	○	◉	○	○
Tonno	○	○	◉	○
Hawaii	○	◉	○	○

- Neue Bestellung

2. Übung: Rohform mit statischem HTML

es geht um
Struktur und Inhalte;
Layout und Formate
sind
Thema der 3. Übung

- Realisieren Sie alle Seiten mit HTML5
  - ⇨ "Rohform": keine physische Formatierung, keine CSS
  - ⇨ verwenden Sie die Codierung UTF-8
  - ⇨ schreiben Sie Umlaute und das €-Zeichen direkt in Ihre Dateien (also nicht als "named character entities / benannte Zeichen" wie ä etc.)
- Werkzeug
  - ⇨ irgendein HTML Editor (z.B. Notepad++), kein WYSIWYG
  - ⇨ achten Sie auf ordentliche Formatierung des HTML-Quelltextes !
  - ⇨ schreiben Sie alle Dateien in UTF-8 ohne BOM (wegen PHP)
  - ⇨ als Ersatz für die fehlende BOM schreiben Sie in jede Datei wenigstens einen Umlaut (ggf. in Kommentar), damit man die Codierung verifizieren kann
- Realisieren Sie zunächst auch diejenigen Seiten statisch, die später dynamisch aus der Datenbank generiert werden sollen
  - ⇨ tragen Sie Beispieldaten ein – so dass klar ist, was später generiert werden muss
  - ⇨ trennen Sie statische und generierte Daten voneinander
- Testen Sie Ihre Seiten mit Firefox UND Internet Explorer und validieren Sie die Standardkonformität

2. Übung: Tipps für die Entwicklung

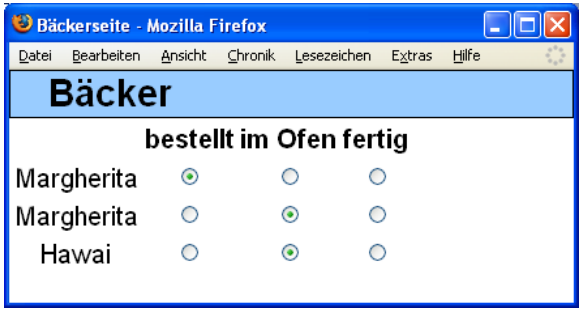
- Umgebung
  - ⇨ Schalten Sie in der Entwicklungsphase im Firefox-Addon "Web Developer" den Browser Cache ab
  - ⇨ Schauen Sie bei Problemen auch einmal in die Fehlerkonsole

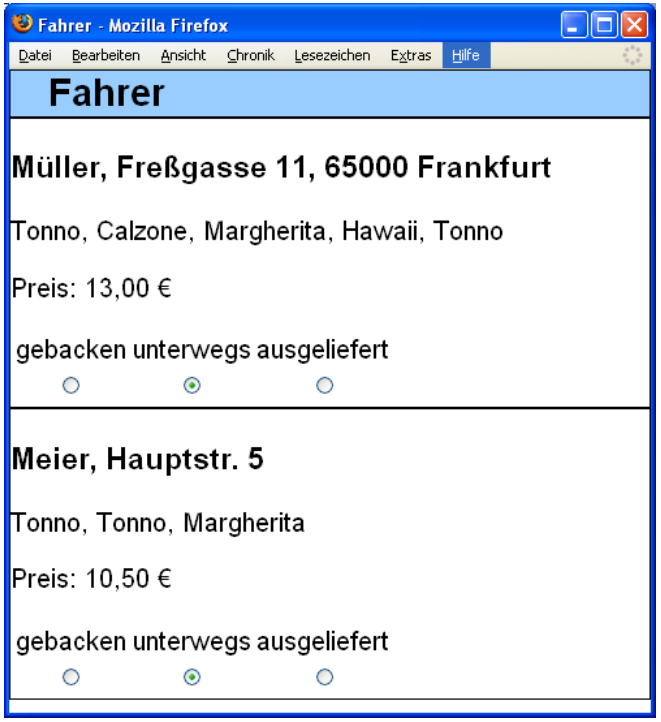
- Formulare
  - ⇨ die Formulare schicken Sie zur Überprüfung an `http://www.fbi.h-da.de/cgi-bin/Echo.pl` (dieses CGI-Skript zeigt die übermittelten Formulardaten an)
  - ⇨ Wenn ein Radiobutton Daten abschicken soll, verwenden Sie Befehle dieser Art (den Unterschied diskutieren wir noch)

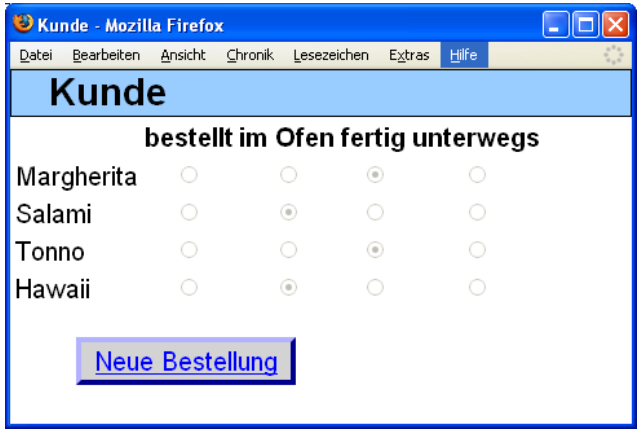
```
<input type="radio" name=...
  onclick="document.forms['formid'].submit();" />        oder
  onclick="window.location.href='Baecker.php?piz=27&amp;stat=f';" />
```

  - ⇨ Beachten Sie, dass die Elemente in einer <select>-Box ausgewählt sein müssen, damit sie übertragen werden

3. Übung: Zielsetzung







3. Übung: Verfeinerung mit CSS und JavaScript (1)

Testen Sie Ihre Seiten mit Firefox und Internet Explorer und validieren Sie HTML und CSS (per Upload an den W3C-Validator)

- Entwickeln Sie ein gemeinsames Style Sheet für Ihre Website und binden Sie es in alle HTML-Seiten ein
  - ⇨ überlegen Sie sich zu diesem Zweck ein Design-Schema
  - ⇨ Farben, Schriftarten, Schriftgrößen, Abstände, Ränder, Ausrichtung, ...
  - ⇨ verwenden Sie keine physische Formatierung in HTML
  - ⇨ verwenden Sie kein style-Attribut
  - ⇨ setzen Sie die verschiedenen Maßeinheiten sinnvoll ein (das Layout soll hinsichtlich Fenstergröße und Schriftgröße dynamisch sein)
- Realisieren Sie das Layout der Seite "Bestellung" barrierefrei mit CSS, nicht mit einer Tabelle
  - ⇨ Tabellen dürfen Sie schon einsetzen, aber nur für wirklich tabellarische Daten (Speisekarte und Statusliste)
  - ⇨ Bedenken Sie, dass die Speisekarte lang werden kann

3. Übung: Verfeinerung mit CSS und JavaScript (2)

Tipp: xxx.toFixed(2) macht aus einer Zahl einen String mit 2 Nachkommastellen und vermeidet Rundungsfehler

- Einsatzbereiche für ECMAScript im Pizzaservice:
  - ⇨ Klick auf ein Pizzabild trägt diese Pizza in den Warenkorb (Liste) ein
    - 3 Pizzen bestellen ⇨ 3-mal klicken
  - ⇨ Möglichkeit zum Löschen der Einträge im Warenkorb
    - sowohl "Alle löschen" als auch "(Mehrfach-)Auswahl löschen"
  - ⇨ MouseOver-Effekte
  - ⇨ Berechnung des Preises
  - ⇨ Klick auf Radio-Button in der Statusliste sendet Statusänderung (bei Bäcker und Fahrer)
- Verwenden Sie nur DOM-konforme Attribute und Funktionen
  - ⇨ keine browserspezifischen Spezialitäten (Vorsicht mit Vorlagen aus dem Netz !)
- Verwenden Sie `"use strict";` in allen Funktionen
- Testen Sie Ihre Skripte mit Firefox und Internet Explorer

3. Übung: Tipps für die Entwicklung

- Umgebung
  - ⇨ Aktivieren Sie Firebug oder den im Browser eingebauten Debugger
  - ⇨ Nutzen Sie den DOM-Inspector und den Befehl "Inspect" im rechten Mausmenü um Probleme in der Webseite im DOM zu lokalisieren
  - ⇨ Achten Sie auf Groß-Klein-Schreibung bei DOM-Aufrufen

- Formulare
  - ⇨ die Formulare schicken Sie zur Überprüfung an `http://www.fbi.h-da.de/cgi-bin/Echo.pl` (dieses Skript zeigt die übermittelten Formulardaten an)
  - ⇨ Achten Sie darauf, dass alle Eingabedaten übermittelt werden !

4. Übung: JavaScript advanced oder Apache Webserver

- Falls Sie mit der Konfiguration eines Webservers vertraut sind, so empfehle ich Ihnen, Ihre JavaScript-Kenntnisse zu vertiefen, d.h. folgende Aufgabenstellung:
  - ⇨ Erweitern Sie den Pizzaservice sinnvoll, z.B.
    - Verwenden Sie bei einer Seite AJAX
    - Fügen Sie beim Fahrer die Visualisierung der optimalen Route mit Hilfe der Google-Api ein
  - ⇨ Implementieren Sie Unit-Tests z.B. für die Berechnung des Gesamtpreises
  - ⇨ Verwenden Sie sinnvolle Design-Patterns, z.B. das Modulpattern http://addyosmani.com/resources/essentialjsdesignpatterns/book/#modulepatternjavascript

- Wer noch nie einen Webserver konfiguriert hat, bearbeitet bitte die Aufgabenstellung der folgenden Folien.

4. Übung: Apache Webserver (1)

Test auf Fehler in httpd.conf:
apache\bin\httpd.exe -t

Starten Sie Apache nach jeder Änderung an der httpd.conf neu!

- Pizzaservice über den Apache ausliefern
  - ⇨ Ihre Pizzaservice-Dateien liegen in einem Verzeichnis in Ihrem Home-Directory oder auf dem Desktop
  - ⇨ Auf den Laborrechnern dürfen Sie nicht die zentrale httpd.conf bearbeiten. Kopieren Sie die `httpd.conf` Datei in Ihr Verzeichnis und starten den Apache durch `httpd.exe -f <myConfig>`
  - ⇨ Konfigurieren Sie den Apache so, dass `localhost\ewa` die Übersichtsseite Ihres Pizzaservices aus obigem Verzeichnis zeigt
    Verwenden Sie dazu `alias` und `directory` (und nicht documentroot)

Sichern Sie Ihre httpd.conf für die nächste Übung !

- GET und POST vergleichen
  - ⇨ Laden Sie das SW-Paket von der Praktikumsseite und entpacken Sie den Inhalt in obiges Verzeichnis.
  - ⇨ Der Aufruf `localhost\ewa\Praktikum4\CgiTestFormular.html` sollte jetzt ein Formular liefern. Das Abschicken des Formulars liefert aber einen Fehler.
  - ⇨ Tragen Sie in `httpd.conf` einen `ScriptAlias` ein und konfigurieren Sie die Berechtigungen, so dass das Formular das Programm `CgiTest.exe` aufruft
  - ⇨ Testen Sie `CgiTestFormular.html` im Browser:
    - welche URL wird jeweils abgeschickt ?
    - worin unterscheiden sich GET und POST beim Reload der Seite ?
  - ⇨ Fazit: für welchen Zweck nehmen Sie welche Methode im Pizzaservice ?

4. Übung: Apache Webserver (2)

- Zugriffsschutz einrichten
  - ⇨ konfigurieren Sie Apache so, dass er .htaccess-Dateien berücksichtigt
  - ⇨ erstellen Sie je eine .htaccess-Datei für Ihre Website und die Unterverzeichnisse aus dem Softwarepaket (public, private, sel-IP und password) mit folgendem Zugriffsschutz:
    - public: alle Zugriffe sind erlaubt
    - private: nur lokale Zugriffe sind erlaubt
    - sel-IP: nur bestimmte IP-Adressen dürfen zugreifen
      - testen Sie den Zugriff in Zusammenarbeit mit Ihren Nachbarn: der rechte darf, der linke nicht
      - Ermitteln Sie Ihre eigene IP-Adresse (mittels `ipconfig` in der DOS-Box) und tauschen Sie sie mit Ihren Nachbarn aus
    - password: Für den Zugriffsschutz mit Passwort erstellen Sie in der DOS-Box eine User/Passwort-Datei mit `<IhrPfadZumApache>\bin\htpasswd.exe`
      - Hilfe zum Aufruf bekommen Sie mit `htpasswd -h`
      - schauen Sie sich die User/Passwort-Datei im Editor an
      - ordnen Sie diese Datei Ihrem Passwort-geschützten Unterverzeichnis zu
      - sorgen Sie dafür, dass die Passwort-Datei keinesfalls vom Webserver ausgeliefert werden kann

4. Übung: Apache Webserver (3)

- MIME-Types
  - ⇨ Die Dateien `MimeTest.html` und `MimeTest.ewa` (aus dem SW-Paket) liegen im Verzeichnis `Praktikum4`
  - ⇨ Rufen Sie `MimeTest.html` über den Webserver auf (also nicht Doppelklicken!): `localhost\ewa\Praktikum4\MimeTest.html`
    - Was macht der Browser, wenn Sie die darin verlinkte Datei aufrufen?
    - Testen Sie verschiedene Browser
  - ⇨ Tragen Sie für die Datei-Endung "ewa" den MIME-Type "application/vnd.ms-excel" ein und starten Sie Apache neu
    - Löschen Sie den Browsercache und rufen Sie die Seite erneut auf.
    - Wie verhält sich der Browser jetzt?
    - Testen Sie verschiedene Browser
  - ⇨ Tragen Sie (temporär) für die Dateiendung "pdf" den gleichen MIME-Type ein und kopieren Sie die .ewa-Datei nach .pdf
    - Rufen Sie die pdf-Datei über den Webserver ab
    - Wird die Datei als PDF oder als Excel-Datei geöffnet?

5. & 6. Übung: Zielsetzung und allgemeine Hinweise

- Zielsetzung
  - ⇨ Sie sollen die Datenbankanbindung mit PHP und MySQLi verstehen
  - ⇨ Sie sollen Objektorientierung mit PHP verstehen und üben
  - ⇨ Dazu implementieren Sie den Pizzaservice unter Verwendung von Klassen und verwalten die anfallenden Daten in einer MySQL Datenbank
- Hinweise
  - ⇨ Alle Seiten müssen objektorientiert unter Verwendung der gegebenen Templates (Download von der EWA-Webseite) implementiert werden
  - ⇨ Alle 4 HTML-Seiten werden unter Verwendung der Daten aus der Datenbank mit PHP erzeugt
  - ⇨ Alle Zugriffe auf die Datenbank erfolgen über MySQLi
  - ⇨ Der Übergang zwischen 5. und 6. Übung ist fließend und bleibt Ihnen überlassen – aber für die Abnahme von Termin 6 muss alles fertig sein !

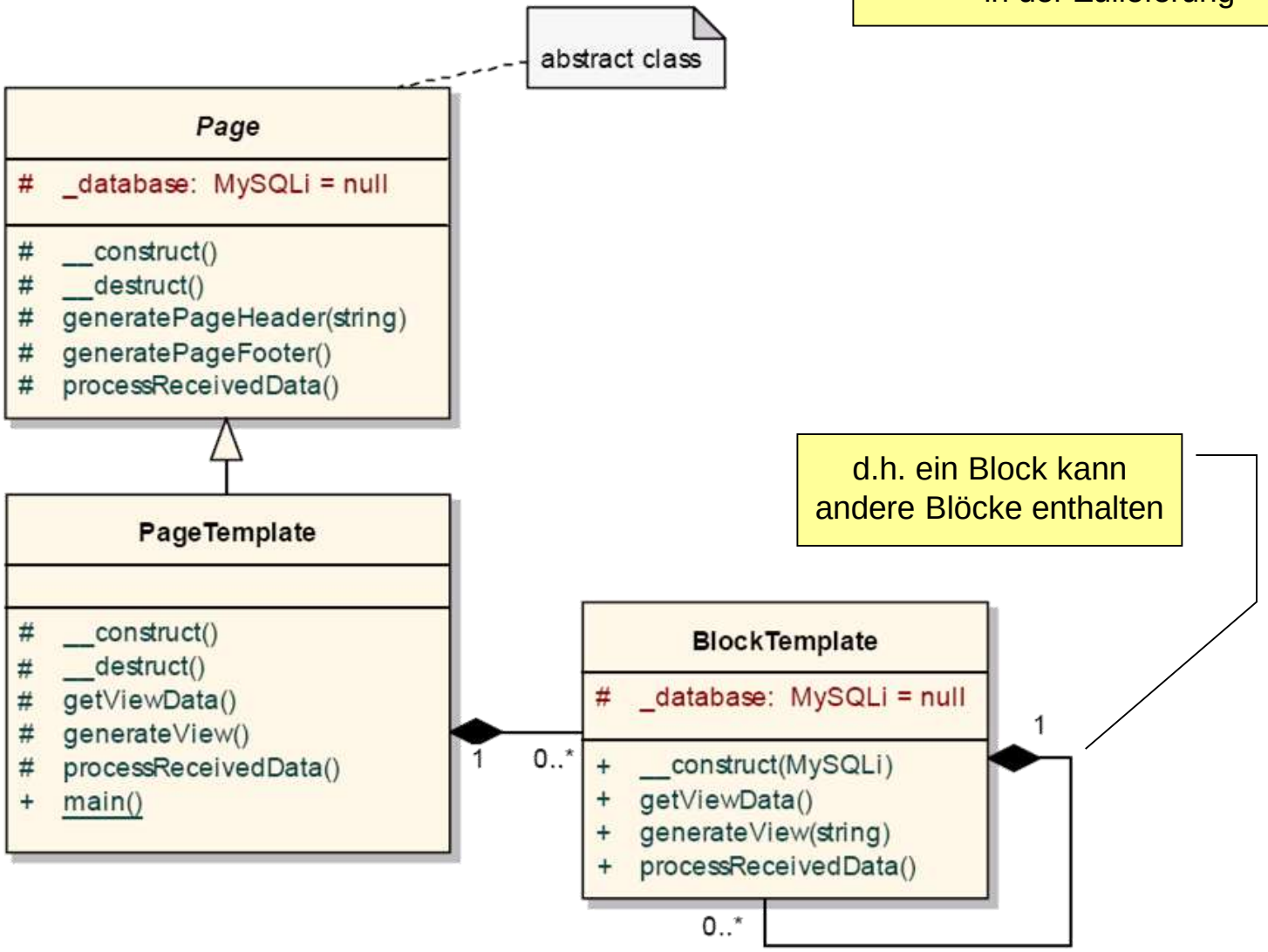
Achtung! Die reine Umsetzung der Funktionalität des Pizzaservice reicht nicht für die Abnahme! Die objektorientierte Umsetzung mit Klassen ist ebenfalls ein Pflichtbestandteil!

5. Übung: Vorbereitung

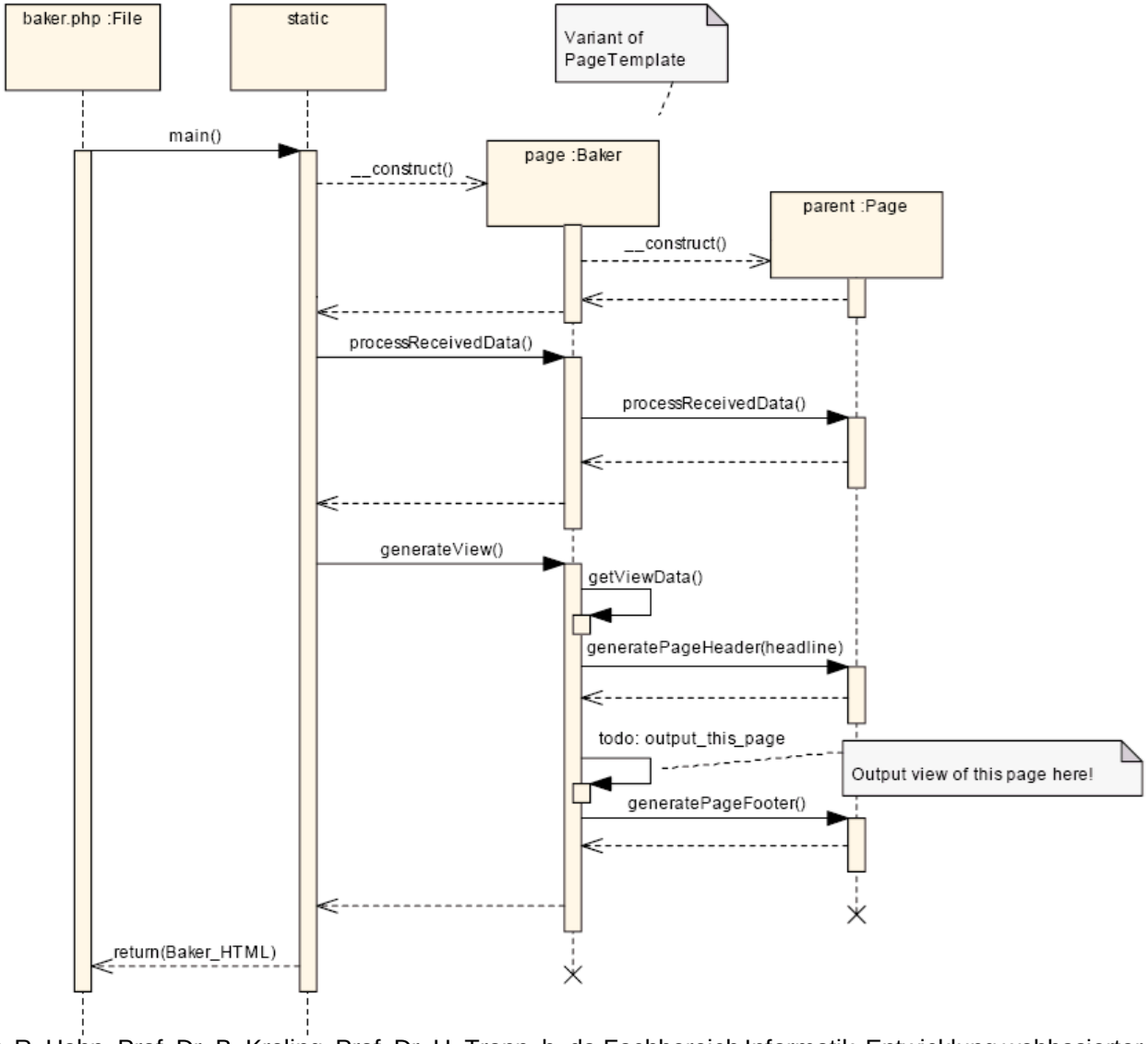
- Vorbereitung
  - ⇨ Stellen Sie Ihre Apache / PHP-Installation aus der 4. Übung wieder her
  - ⇨ Laden Sie die Zulieferung für das Praktikum von der Webseite herunter:
    - PageTemplate.php dient als Vorlage für die 4 Seiten Bestellung.php, Kunde.php, Baecker.php und Fahrer.php
    - Page.php ist die gemeinsame Basisklasse dieser 4 Seiten-Klassen und soll die Datenbank öffnen und schließen und den HTML-Rahmen erzeugen
    - BlockTemplate.php dient als Vorlage für einzelne Blöcke innerhalb der Seiten
    - Pizzaservice_Documentation.pdf enthält eine Dokumentation der Klassen mit Klassendiagramm und Sequenzdiagramm
  - ⇨ Versuchen Sie mit der Dokumentation und dem Quellcode das Zusammenspiel der verschiedenen Klassen zu verstehen. Klären Sie folgende Fragen:
    - Wo erfolgt der eigentliche Aufruf zur Erstellung einer HTML-Seite?
    - Was tun die Methoden getViewData(), generateView() und processReceivedData()?
    - Wo wird der HTML-Rahmen erzeugt? Wo wird er ausgegeben?
  - ⇨ Legen Sie 4 Kopien der Klasse PageTemplate.php an und nennen Sie die Dateien Bestellung, Status, Baecker und Fahrer. Ändern Sie auch die Klassennamen und Verweise auf die Klasse innerhalb der Dateien.

5. Übung: Klassendiagramm der Templates

siehe auch Pizzaservice_Documentation.pdf in der Zulieferung



5. Übung: Sequenzdiagramm der Templates



5. Übung: HTML in Klassen übertragen

- Vorgehen
  - ⇨ verteilen Sie Ihren HTML-Code aus der 3. Übung in die zuständigen Methoden der 5 Klassen Page, Bestellung, Status, Baecker und Fahrer
    - die Kommentare in den Dateien helfen dabei
    - HTML-Ausgaben erfolgen nur in generateView() !
    - die Struktur wird besser, wenn Sie Speisekarte, Warenkorb-Formular und Statustabellen unter Verwendung von BlockTemplate.php realisieren (das ist aber optional)
    - die Ausgabe größerer HTML-Abschnitte ist mit der "Heredoc-Notation" besonders einfach. Achten Sie darauf, dass die Endmarke in der ersten Spalte beginnen muss und höchstens noch ein ; folgen darf
  - ⇨ schreiben Sie Hilfs-Methoden oder zusätzliche Klassen nach Bedarf
    - die Methoden der gegebenen Klassen dürfen auch zusätzliche Parameter bekommen
- Test
  - ⇨ Prüfen Sie, ob die neuen PHP-Seiten als Ausgabe die ursprünglichen statischen HTML-Seiten erzeugen
  - ⇨ Wird der erzeugte HTML-Code immer noch vom Validator akzeptiert?
  - ⇨ Wenn die Tests erfolgreich sind, können Sie mit der Implementierung der DB-Zugriffe beginnen

5. Übung: Implementierung der Datenbank

Bis hierher sollten Sie in der 5. Übung mindestens kommen!

- Entwerfen Sie das Datenmodell für den Pizzaservice, z.B.
  - ⇨ Angebot: PizzaName, Bilddatei, Preis
  - ⇨ BestelltePizza: PizzaID, fBestellungID, fPizzaName, Status
  - ⇨ Bestellung: BestellungID, Adresse, Bestellzeitpunkt
- Implementieren Sie das Datenmodell mit phpMyAdmin
  - ⇨ verwenden Sie die Kollation utf8_unicode_ci (im Vorgabewert utf8_general_ci gilt nicht ß=ss)
  - ⇨ PizzaName, PizzaID, BestellungID sind Primärschlüssel; IDs mit Autoincrement
  - ⇨ realisieren Sie die Verknüpfungen zwischen den Primärschlüsseln und den Fremdschlüsseln fBestellungID, fPizzaName in der Datenbank
    Tipp: Mit dem "Designer" in phpMyAdmin können Sie die Beziehungen grafisch eintragen
  - ⇨ füllen Sie die Tabelle "Angebot" manuell mit phpMyAdmin
- Tipp zum Bestellzeitpunkt
  - ⇨ MySQL-Funktion CURRENT_TIMESTAMP als Standardwert des Feldes

6. Übung: Datenbankanbindung mit PHP / MySQLi

Alle Seiten müssen objektorientiert unter Verwendung der gegebenen Templates implementiert werden

- Vorgehensweise
  - ⇨ implementieren Sie die Datenbankzugriffe (Select, Insert Into, Update) in den zuständigen Methoden der Klassen
    - der Zugriff auf die Datenbank erfolgt objektorientiert über die Klasse MySQLi
    - Zugriff auf die Datenbank erfolgt nur in getViewData() und processReceivedData()
  - ⇨ ersetzen Sie die bisherigen statischen Tabellen durch PHP-Code, der die Zeilen aus den abgefragten Daten generiert
    - bilden Sie den bisherigen statischen HTML-Code exakt nach !
    - schreiben Sie Hilfs-Methoden oder zusätzliche Klassen nach Bedarf
  - ⇨ testen und debuggen ... error_reporting(E_ALL) hilft dabei
- Tipps zur Umsetzung
  - ⇨ var_dump($variable) für die schnelle Testausgabe zwischendurch
  - ⇨ number_format($zahl, $nachkommastellen) formatiert $zahl
  - ⇨ $mysqli->insert_id liefert die Autoincrement-ID nach INSERT INTO
  - ⇨ Tabellen- und Feldnamen in MySQL ggf. in ` (Gravis / accent grave) einklammern
  - ⇨ prüfen Sie mit phpMyAdmin ob die Datenbankeinträge korrekt erstellt werden

6. Übung: Sessionverwaltung und Sicherheit

- Der Kunde soll auf seiner Statusseite nur diejenigen Pizzen sehen, die er selbst zuletzt bestellt hat
  - ⇨ Implementieren Sie dieses Feature mittels Sessionverwaltung: speichern Sie die letzte AuftragsNr in der Session und filtern Sie damit die Pizzaliste
- Verhindern Sie SQL-Injection mit Hilfe von real_escape_string
  - ⇨ Test: geben Sie / ' " \ als Lieferadresse ein; diese Zeichen müssen auf der Fahrerseite genau so erscheinen
- Verhindern Sie Cross Site Scripting mit Hilfe von htmlspecialchars
  - ⇨ Test: geben Sie xxx als Lieferadresse ein; dies muss genau so in der Datenbank und in der Ausgabe auf der Fahrerseite erscheinen
- abschließend testen und generierte Seiten validieren
- Abnahme machen