

# Dokumentation

Felix Krautschuk (s68334)

04.07.2014

# Contents

1. Theoretische Einordnung
2. Dokumentation in unserem Projekt
  - 2.1. Meine Aufgabe
  - 2.2. Verwendetes Werkzeug
  - 2.3. Protokollieren der Teambesprechungen
  - 2.4. Pflichtenheft
  - 2.5. Entwicklerdokumentation
  - 2.6. Benutzerdokumentation
  - 2.7. Testdokumentation
  - 2.8. Reflexion (Dokumentation)
3. Gesamteinschätzung

# 1. Theoretische Einordnung

- Fragen:
  - Welche Bedeutung hat die Dokumentation für ein Software-Projekt?
  - Wer sollte die Dokumentation verfassen?
  - Welche Dokumentation für welche Zielgruppe?
  - Wie gestaltet man eine Dokumentation ansprechend und verständlich?
  - Wann ist eine Dokumentation endgültig fertig?

# 1. Theoretische Einordnung

Welche Bedeutung hat die Dokumentation für ein Software-Projekt und welchen Umfang sollte sie haben?

## Ziele:

- *Klarheit* über Problemstellung schaffen
  - Anforderungsanalyse → Pflichtenheft
  - sonst kann eigentliche Arbeit nicht beginnen
- sich *möglichst schnell* mit *wenig Aufwand* in ein bestehendes System einarbeiten
  - Entwicklerdokumentation
- Fehlerbehandlung
  - Testdokumentation mit Testberichten
- Software schnell und einfach *bedienen*
  - Benutzerdokumentation

# 1. Theoretische Einordnung

Wer sollte die Dokumentation verfassen?

## **Sinnvoll:**

- Spezialist des jeweiligen Gebietes
  - aus thematischer Sachlage betrachtet
  - hat am meisten Hintergrundwissen
  - kein erneutes „Hineindenken“ durch fremde Person erforderlich

## **Probleme:**

- Dokumentationserstellung meist nur zweitrangig
- Projektleiter hat Aufgabe an jemanden delegiert
- Termindruck
- Qualität leidet unter Umständen darunter

## 2. Dokumentation in unserem Projekt

- Was war also meine Aufgabe?
  - Zusammenarbeit mit Experten/Spezialisten der einzelnen Bereiche im Software-Entwicklungsprozess
  - deren Ergebnisse zusammentragen
  - in ansprechender Form dokumentieren
    - Bezug auf Zielgruppen
- Zusätzlich:
  - Protokollierung der Gruppensitzungen
  - Verfassung einer Projektdokumentation

## 2.2. Verwendetes Werkzeug



+



=

TeXlipse

## 2.3. Protokolle

- Dokumentation der Teambesprechungen
- Ziel:
  - Was war unser Ziel? (Tagesordnung)
  - Was wurde erreicht?
  - Wer erarbeitet was bis zum nächsten Treffen?
- Vorgehensweise:
  - zunächst „schnelle“ Mitschrift in Textdatei
  - Erstellen eines strukturierten Protokolls in Latex
  - Bereitstellung in Github
- Hilfsmittel:
  - Audio-Mitschnitt

### 6.2 2. Treffen

Termin: 23.04.2014 in der Bibliothek B302a

Tagesordnung:

- erste Ideen für die Analyse der Problemstellung zusammentragen
- Klarheit über Komplexität und Detailliertheit der Features des Systems
- Entwurfsmöglichkeiten diskutieren verschaffen

Probleme die diskutiert wurden, aber noch mit Frau Hauptmann zu besprechen sind:

Wie sollen die Studenten-Gruppen überhaupt verwaltet werden:

- 1. Möglichkeit: Dozent trägt die Mitglieder der Gruppe selbst in das System ein
  - Dozent hat einen guten Überblick über alle Gruppen und sieht sofort, welche Gruppen bereits vollständig sind und wo noch Mitglieder fehlen und welche Studenten noch keiner Gruppe zugeordnet wurden
  - aber: dadurch hat der Dozent am meisten Aufwand bei der Organisation der Gruppen
- 2. Möglichkeit: Dozent erstellt für die Gruppen eine Art Grundgerüst (Rahmen) und die Studenten tragen dann in diese zunächst leeren Gruppen ein
  - Dozent bleibt beim Eintragen der Gruppen passiv, hat nur Kontrollfunktion
  - müsste das Eintragen dann online (in einer Webanwendung) geschehen??
  - wäre schwieriger zu implementieren als 1. Variante
- 3. Möglichkeit: potentielle Gruppenleiter melden sich bei Dozenten und werden eingetragen
  - diese suchen sich ihre Mitglieder selbst
  - es gibt somit keine leeren Gruppen
  - Organisation der Gruppen geschieht ebenfalls durch Studenten selbst

Filtermöglichkeiten??

wie komplex soll das System insgesamt werden?? Viele Features oder einfach halten??

Was umfasst der Entwurf alles??

---

Plan für nächstes Treffen:

Termin: 30.04.2014

Benjamin Reim: Dialoge, eventuell erste Entwürfe

Markus Noack: Archivierung, alles was zum Schluss aus der Datenbank in der PDF-Datei stehen muss



## 2.3. Protokolle

Erfahrungen:

- Schwierig, immer konzentriert mitzuschreiben (bei Diskussionen)
- Was ist wichtig, was nicht?

Gute Idee:

- Tonmitschnitte (wenn auch nur selten benötigt)
- Herangehensweise (Mitschriften → strukturiertes Protokoll)

## 2.4. Pflichtenheft

„Wer sollte die Dokumentation verfassen?“

- Spezialist in jeweiligen Arbeitsbereich
- Verantwortlicher für Anforderungsanalyse (Martin) hat Aufgaben verteilt
  - Ergebnisse wurden zusammengetragen und von Martin ordentlich ausformuliert
  - sonst keine explizite Zusammenarbeit

## 2.5. Entwicklerdokumentation

### Vorgehensweise

- Entwurf und Architektur
  - Zusammenarbeit mit Benjamin Reim
- Implementierung
  - Zusammenarbeit mit Benjamin Herzog und Markus Noack

### Ziel:

grober Überblick  Blick ins Detail

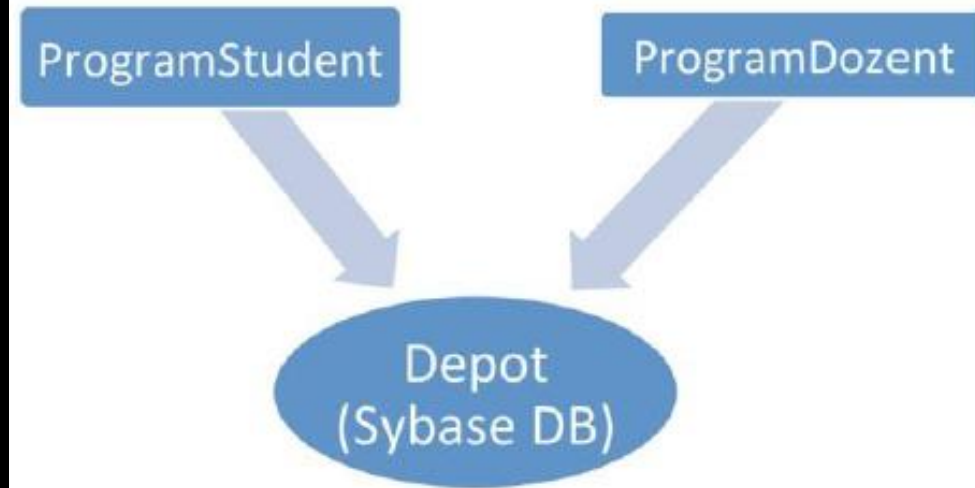
## 2.5. Entwicklerdokumentation

- **1. Entwurf**
  - Architektur
    - Erläuterung mittels Text
    - Veranschaulichung durch Grafik

### 2.1 Entwurfsarchitektur

Bei der Wahl der Entwurfsarchitektur konnte auch auf Erfahrungen des Praktikums des Modules Software Engineering II zurückgegriffen werden. Aufgrund der Eigenart des zu entwickelnden Softwaresystems fiel die primär auf die Depot-Architektur, wobei diese mit der 3-Schichten Architektur kombiniert ist.

#### 2.1.1 Depot-Architektur

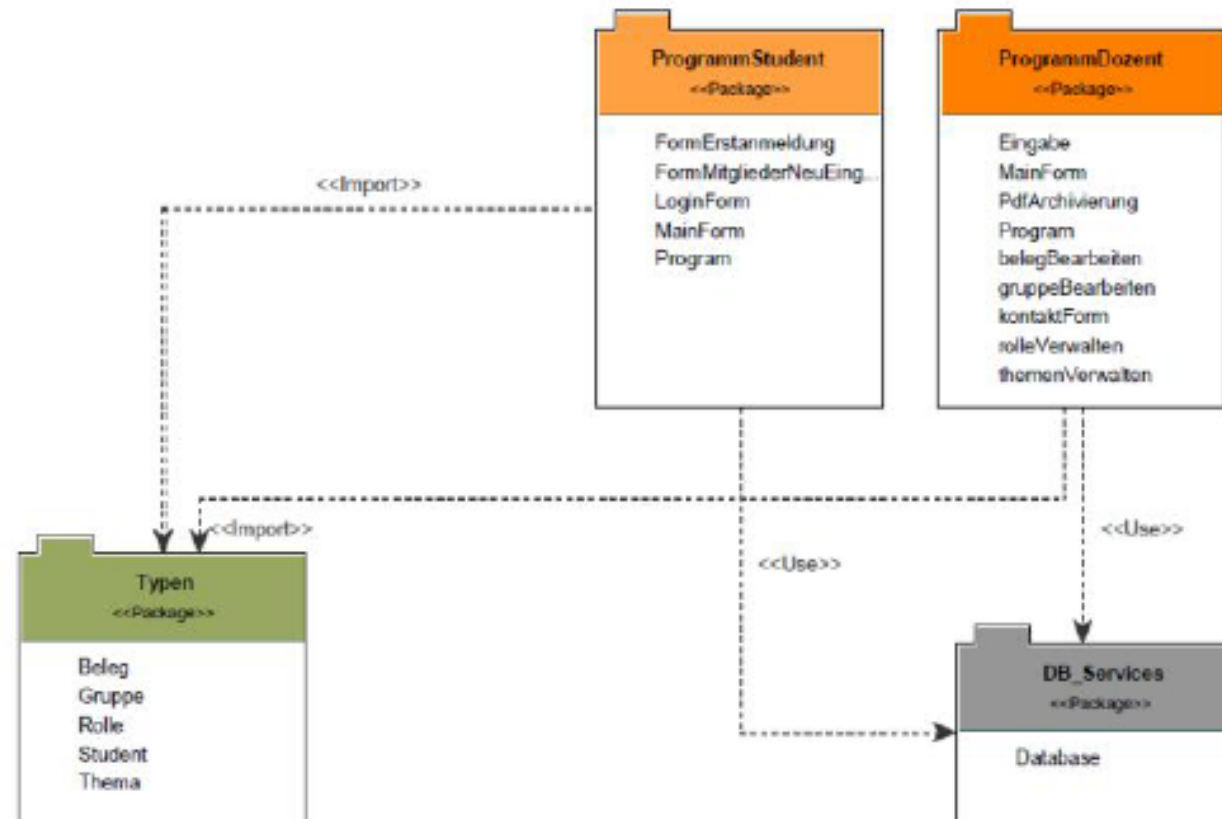


Mit der Depot-Architektur steht ein zentrales Depot (Repository) bereit, auf das von mehreren Subsystemen zugegriffen wird. Das Depot besteht in dem vorliegenden Softwareprojekt aus einer Sybase-Datenbank auf die mit dem Programmen Dozent bzw. Student zugegriffen wird. (siehe Abbildung)

## 2.5. Entwicklerdokumentation

- **1. Entwurf**
  - Paketstruktur
    - Paketdiagramm
    - textliche Erläuterungen

### 2.2 Paketdiagramm



Wie aus dem Paketdiagramm ersichtlich ist ist der Code aufgetrennt in eine Logikschicht (Packet Typen), der Datenhaltungsschicht (Packet Database\_Services) sowie der Präsentationsschicht (Pakete ProgramDozentUI und ProgramStudentUI). Mit einer Trennung der Schichten wird eine klare Struktur, hohe Kohäsion & lose Kopplung (und damit mit Portabilität) sowie eine bessere Zugriffskontrolle erreicht.

# 2.5. Entwicklerdokumentation

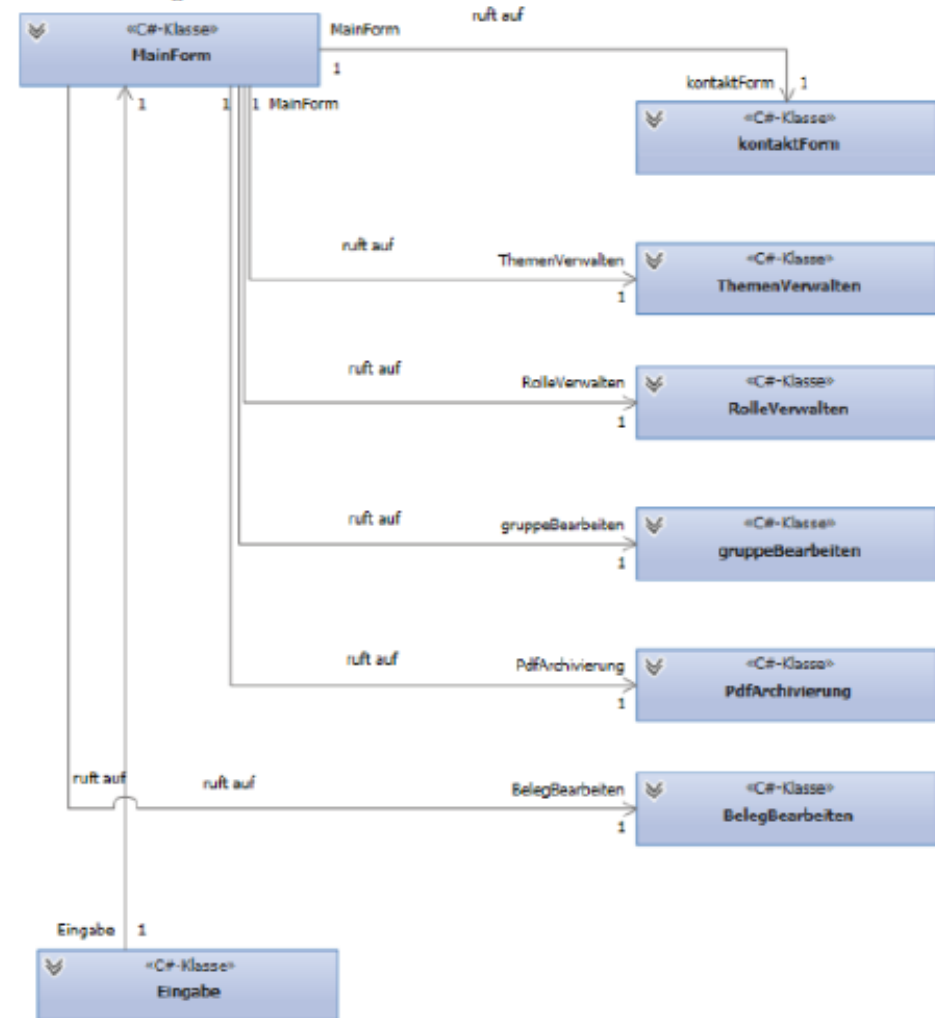
- **1. Entwurf**
  - Klassenstruktur
    - Klassendiagramme für
      - Typen
      - Präsentationsschicht (Dozentenprogramm und Studentenprogramm)
      - Datenbank
    - Jeweils mit textlicher Beschreibung

Hier: logische Aufrufhierarchie der Dialoge

## 2.3.2 Präsentationsschicht

Um den dokumentierende Charakter gerecht zu werden und ein gesundes Maß an Granularität zu gewährleisten wurde darauf verzichtet die Klassen der Präsentationsschicht mit allen Labeln etc. darzustellen. Stattdessen ist in den unteren Diagrammen die logische Aufrufhierarchie innerhalb der unterschiedlichen Dialoge dargestellt.

Dozenten-Programm:

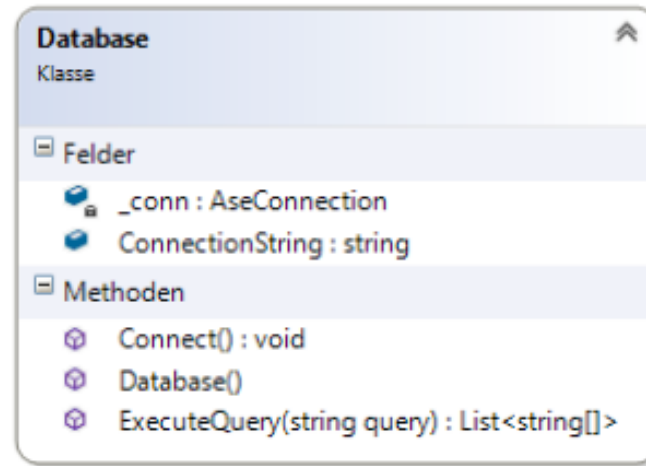


## 2.5. Entwicklerdokumentation

- 2. Implementierung
  - Paket
    - Klasse
      - Klassendiagramm (ohne Methoden)
      - Allgemeine Beschreibung
      - Funktionsprototyp mit textlicher Beschreibung

### 3.4 Paket DBService

#### 3.4.1 Die Klasse Database



Diese Klasse stellt Methoden zur Verbindung mit der Datenbank und zur Ausführung von SQL-Statements bereit.

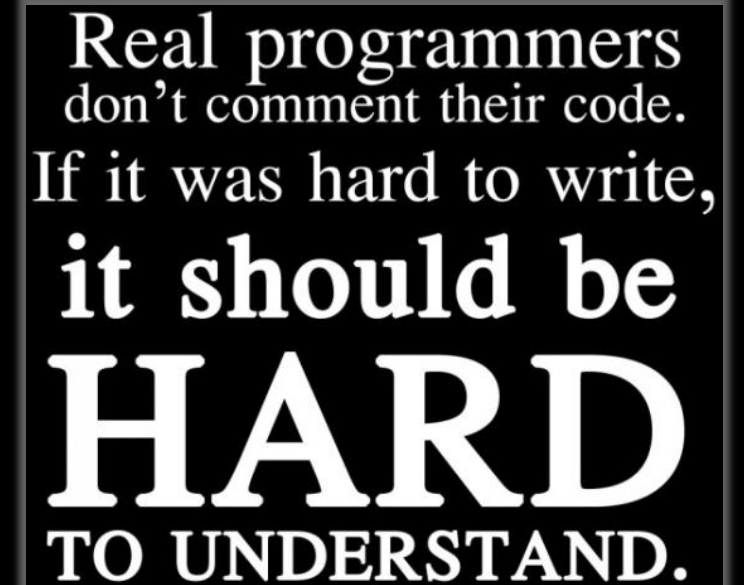
- **public Database()**  
Im Konstruktor wird die Datenbank festgelegt, mit welcher gearbeitet werden soll.
- **public void Connect()**  
Mit dieser Funktion wird die Verbindung zu der Datenbank realisiert bzw eine Fehlermeldung ausgegeben, falls die Verbindung fehlschlägt. Mithilfe des Connectionstrings wird der Datenbanzugriff genauer spezifiziert.
- **public List<string[]> ExecuteQuery(string query)**  
Mithilfe dieser Funktion wird eine Datenbankabfrage ausgeführt und das Abfrageergebnis als Liste von Strings zurückgegeben. Das Ergebnis der Abfrage wird zunächst in der Variable datareader gespeichert. In der while-Schleife wird daraus nun die String-Liste erstellt.

## 2.5. Entwicklerdokumentation

### Erfahrungen

umfangreichstes Dokument

- Positiv:
  - allgemein gute Zusammenarbeit mit Experten aus „Entwurf“ und „Implementierung“
    - Arbeitsteilung
- Schwierigkeiten:
  - selbstständiges Arbeiten meinerseits nur bedingt möglich
- Verbesserung:
  - Programmierer mehr bei Kommentierung überwachen
    - aus ihrer Sicht genügend Kommentare



Real programmers  
don't comment their code.  
If it was hard to write,  
**it should be**  
**HARD**  
TO UNDERSTAND.



## 2.6. Benutzerdokumentation

### Ziel:

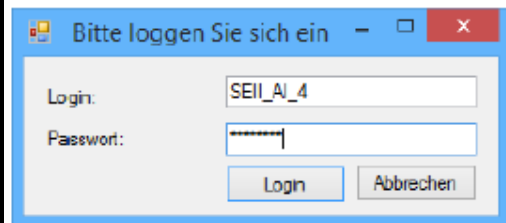
- Studenten und Dozent bei der Verwendung ihres jeweiligen Programmes zu unterstützen
  - Dozent beim Verwalten der Belege
  - Student beim Anmeldevorgang

## 2.6. Benutzerdokumentation

### Formen der Dokumentation

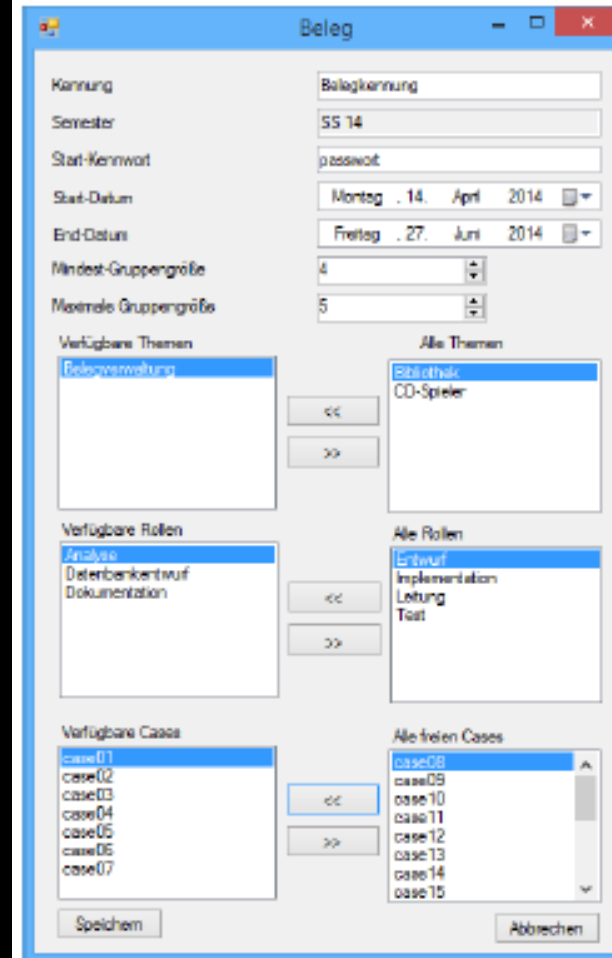
- Textliche Erläuterung der einzelnen Menüs (einfache Formulierungen)
- Unterstützung durch Screenshots mit Daten
- Auf mögliche Fehler bei den Eingaben wird nicht eingegangen!

### 3 Erstanmeldung



Beim Programm-Start wird der sich einloggende Student aufgefordert die Gruppenkennung und das Gruppenpasswort einzugeben. Falls es sich hierbei um eine Erstanmeldung handelt, müssen nun die vom Dozenten bekanntgegebenen Erstlogin-Daten eingegeben werden.

### 4.1 Menü: Beleg anlegen



Dieses Menü erlaubt es einen neuen Beleg für das aktuelle Semester, welches in der zweiten Zeile angezeigt wird, anzulegen. Der Dozent kann hier das Startpasswort für den Beleg, sowie den Zeitraum und die Gruppengröße festlegen. Mit diesem Startpasswort können die Studenten später ihre Gruppen anlegen und diese mit Mitgliedern füllen.

Im unteren Bereich dieses Fensters können speziell für diesen Beleg die wählbaren Themen und die festzulegenden Rollen/Verantwortlichkeiten ausgewählt werden, sowie die

## 2.7 Testdokumentation

- geringes „Einmischen“ meinerseits
  - Template vorbereitet
  - ansonsten hat Tester hat selbstständig gearbeitet
  - ich hatte keine Verbesserungsvorschläge

## 2.8. Reflexion (Dokumentation)

- Dokumentation wieder in Latex
- gute Zusammenarbeit (Entwicklerdokumentation)
  - aber: Zusammenarbeit mit den anderen Verantwortlichen von Beginn an!
- Teilweise zu wenig Verantwortung übernommen (Pflichtenheft, Testdokumentation)

### 3. Gesamteinschätzung (Projekt)

- Interessant, einmal den kompletten SW-Entwicklungsprozess geübt zu haben
  - neben Dokumentation: Programmierung der PDF-Archivierung (Dozentenprogramm)
    - Zusammenarbeit mit Christian Knothe
      - ➡ Erfahrungen in mehreren Bereichen gesammelt
- gute Gruppenarbeit (wir kannten uns gegenseitig bereits)
- leider: zunehmender Stress wegen anderen Studienfächern