

Softwaretechnik 1 - 1. Tutorium

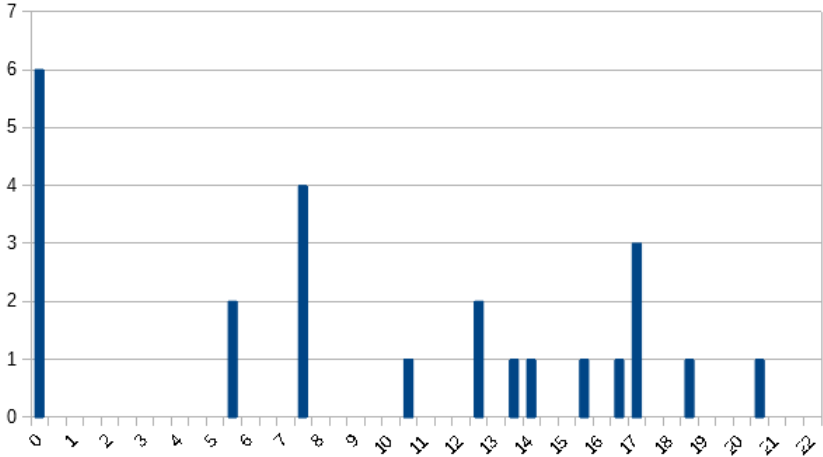
Tutorium 17

Felix Bachmann | 14.05.2019

KIT - INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION (IPD)



1. Übungsblatt Statistik



Allgemein

generell ohne Abzug:

- gleiche Abgabe bei allen Aufgaben

Allgemein

generell ohne Abzug:

- gleiche Abgabe bei allen Aufgaben

generell mit Abzug: (bis zu -2P)

- CheckStyle nicht beachtet
- JavaDoc !(vollständig && sinnvoll)
- Commits !(regelmäßig && aussagekräftig)

Aufgabe 1 (Altsoftware vorbereiten)

- vorgegebene .gitignore verwenden (mit IDE-Zeug), nicht nur target/

Aufgabe 1 (Altsoftware vorbereiten)

- vorgegebene .gitignore verwenden (mit IDE-Zeug), nicht nur target/
- fully.qualified.MainClass durch Paket-Struktur ersetzen (org.jis.Main)

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen "pointer" auf einen Pfad (siehe File.createNewFile() oder File.mkdir())

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen “pointer” auf einen Pfad (siehe File.createNewFile() oder File.mkdir())
- fügt Abhängigkeiten in die jmjrst.main-pom.xml ein, **nicht** in die von iMage

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen "pointer" auf einen Pfad (siehe File.createNewFile() oder File.mkdir())
- fügt Abhängigkeiten in die jmjrst.main-pom.xml ein, **nicht** in die von iMage
- @Test(expected=XYException.class) nutzen

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen "pointer" auf einen Pfad (siehe File.createNewFile() oder File.mkdir())
- fügt Abhängigkeiten in die jmjrst.main-pom.xml ein, **nicht** in die von iMage
- @Test(expected=XYException.class) nutzen
- JUnit4 benutzen

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen "pointer" auf einen Pfad (siehe File.createNewFile() oder File.mkdir())
- fügt Abhängigkeiten in die jmjrst.main-pom.xml ein, **nicht** in die von iMage
- @Test(expected=XYException.class) nutzen
- JUnit4 benutzen
- nicht throws Exception angewöhnen

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen "pointer" auf einen Pfad (siehe File.createNewFile() oder File.mkdir())
- fügt Abhängigkeiten in die jmjrst.main-pom.xml ein, **nicht** in die von iMage
- @Test(expected=XYException.class) nutzen
- JUnit4 benutzen
- nicht throws Exception angewöhnen
- nicht Korrektheit der zu testenden Methoden annehmen

Aufgabe 2 + 3 (Modultests + Testüberdeckung)

- auch bei Drehung um 0° ist Überprüfung des Bildes nötig (Dimensionen + Pixel)
- equals() reicht nicht aus, um Gleichheit der Bilder zu prüfen
- new File() erstellt kein File, sondern nur einen "pointer" auf einen Pfad (siehe File.createNewFile() oder File.mkdir())
- fügt Abhängigkeiten in die jmjrst.main-pom.xml ein, **nicht** in die von iMage
- @Test(expected=XYException.class) nutzen
- JUnit4 benutzen
- nicht throws Exception angewöhnen
- nicht Korrektheit der zu testenden Methoden annehmen
- Stil: Konstanten für Pfade benutzen

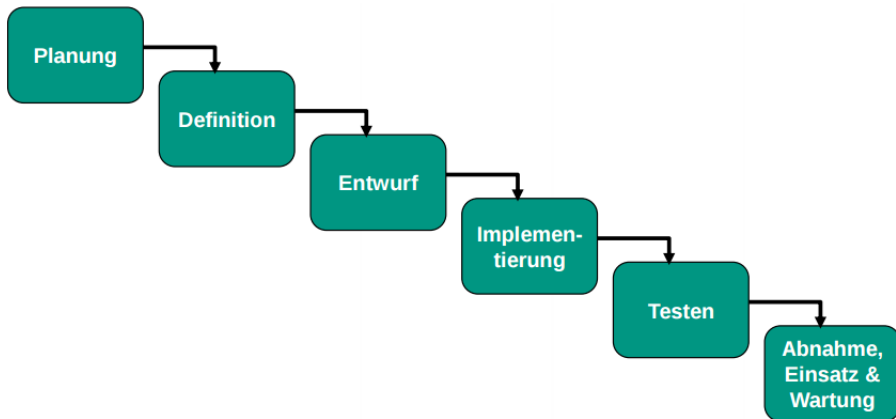
- 1 Orga
- 2 Wasserfallmodell
- 3 Durchführbarkeitsuntersuchung
- 4 Lastenheft
- 5 Pflichtenheft
- 6 UML-Klassendiagramm
- 7 \LaTeX
- 8 Tipps

■ Was ist das?

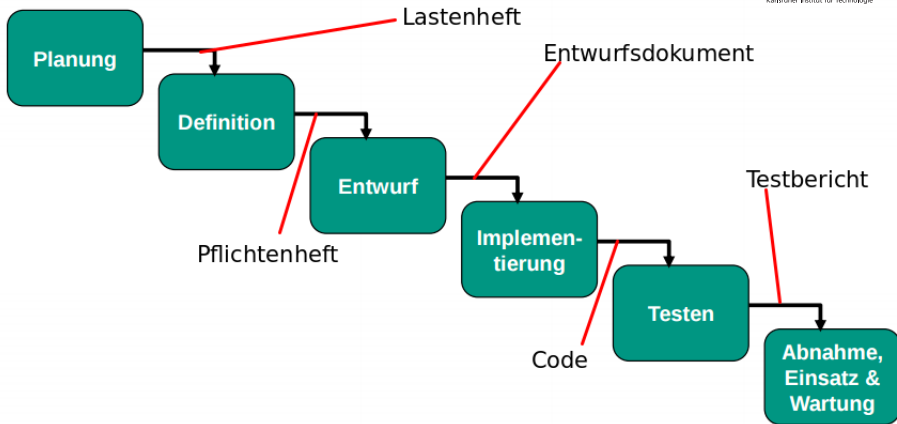
- dokumentengetriebenes Prozessmodell

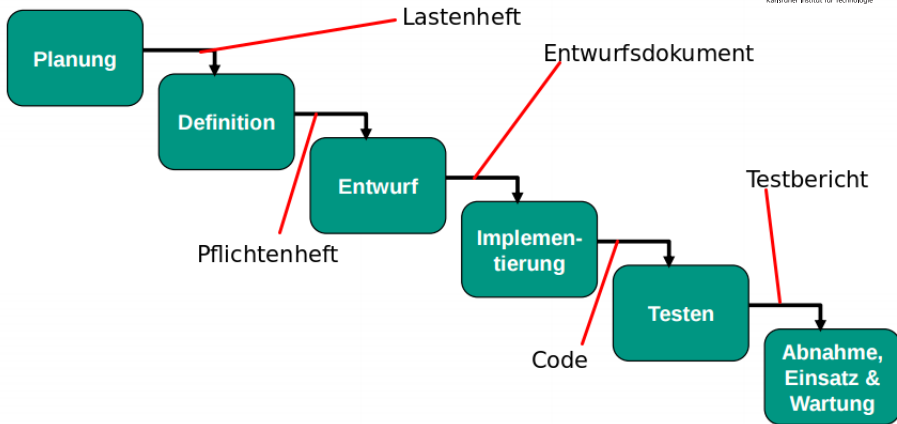
- dokumentengetriebenes Prozessmodell
- mögliche Phasen der Softwareentwicklung

- dokumentengetriebenes Prozessmodell
- mögliche Phasen der Softwareentwicklung



Wasserfallmodell





Dokumente für das 2. ÜB:

- Lastenheft
- Durchführbarkeitsuntersuchung (weiteres Artefakt der Planung)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

1 Fachlich

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken (Gibt es Risiken? :D)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken (Gibt es Risiken? :D)
- 5 Ökonomisch

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken (Gibt es Risiken? :D)
- 5 Ökonomisch (wirtschaftlich? Termine?)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken (Gibt es Risiken? :D)
- 5 Ökonomisch (wirtschaftlich? Termine?)
- 6 Rechtlich

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken (Gibt es Risiken? :D)
- 5 Ökonomisch (wirtschaftlich? Termine?)
- 6 Rechtlich (Datenschutz, Standards)

Grundlegende Frage

Ist das Projekt in dem jeweiligen Szenario überhaupt durchführbar?

- 1 Fachlich (softwaretechnisch leicht realisierbar?)
- 2 Alternativen (lieber altes Projekt anpassen oder komplett neu entwickeln?)
- 3 Personell (genug qualifiziertes Personal?)
- 4 Risiken (Gibt es Risiken? :D)
- 5 Ökonomisch (wirtschaftlich? Termine?)
- 6 Rechtlich (Datenschutz, Standards)

Fürs Übungsblatt

Denkt euch was (plausibles) aus!

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- 1 Zielbestimmung (grobe Beschreibung)

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- 1 Zielbestimmung (grobe Beschreibung)
- 2 Produkteinsatz (Für wen? Zielgruppe, Anwendungsbereich)

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- 1 Zielbestimmung (grobe Beschreibung)
- 2 Produkteinsatz (Für wen? Zielgruppe, Anwendungsbereich)
- 3 Funktionale Anforderungen (feingranular: Funktionen des Produkts)

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- 1 Zielbestimmung (grobe Beschreibung)
- 2 Produkteinsatz (Für wen? Zielgruppe, Anwendungsbereich)
- 3 Funktionale Anforderungen (feingranular: Funktionen des Produkts)
- 4 Produktdaten (Welche Daten speichern?)

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- 1 Zielbestimmung (grobe Beschreibung)
- 2 Produkteinsatz (Für wen? Zielgruppe, Anwendungsbereich)
- 3 Funktionale Anforderungen (feingranular: Funktionen des Produkts)
- 4 Produktdaten (Welche Daten speichern?)
- 5 Nichtfunktionale Anforderungen (Meta-Anforderungen: Zeit, Zuverlässigkeit)

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- ① Zielbestimmung (grobe Beschreibung)
- ② Produkteinsatz (Für wen? Zielgruppe, Anwendungsbereich)
- ③ Funktionale Anforderungen (feingranular: Funktionen des Produkts)
- ④ Produktdaten (Welche Daten speichern?)
- ⑤ Nichtfunktionale Anforderungen (Meta-Anforderungen: Zeit, Zuverlässigkeit)
- ⑥ Systemmodelle
 - Szenarien (spezielles Beispiel)
 - Anwendungsfälle (allgemeiner Verwendungszweck)

Grundlegende Aufgabe

Das Lastenheft sammelt die Anforderungen des Auftraggebers an den Auftragnehmer. Theoretisch vom Kunden geschrieben.

- ① Zielbestimmung (grobe Beschreibung)
- ② Produkteinsatz (Für wen? Zielgruppe, Anwendungsbereich)
- ③ Funktionale Anforderungen (feingranular: Funktionen des Produkts)
- ④ Produktdaten (Welche Daten speichern?)
- ⑤ Nichtfunktionale Anforderungen (Meta-Anforderungen: Zeit, Zuverlässigkeit)
- ⑥ Systemmodelle
 - Szenarien (spezielles Beispiel)
 - Anwendungsfälle (allgemeiner Verwendungszweck)
- ⑦ Glossar (technische Begriffe erklären)

Zielbestimmung vs. Funktionale Anforderungen

Zielbestimmung vs. Funktionale Anforderungen

- Zielbestimmung: allgemeine Beschreibung, was das Produkt können soll
- Funktionale Anforderungen: konkrete Auflistung von Funktionen

Zielbestimmung vs. Funktionale Anforderungen

- Zielbestimmung: allgemeine Beschreibung, was das Produkt können soll
- Funktionale Anforderungen: konkrete Auflistung von Funktionen

Funktionale Anforderungen vs. Nichtfunktionale Anforderungen

Zielbestimmung vs. Funktionale Anforderungen

- Zielbestimmung: allgemeine Beschreibung, was das Produkt können soll
- Funktionale Anforderungen: konkrete Auflistung von Funktionen

Funktionale Anforderungen vs. Nichtfunktionale Anforderungen

- Funktionale Anforderungen: Funktionen des Produkts
- Nichtfunktionale Anforderungen: "Meta"-Eigenschaften des Produkts

Zielbestimmung vs. Funktionale Anforderungen

- Zielbestimmung: allgemeine Beschreibung, was das Produkt können soll
- Funktionale Anforderungen: konkrete Auflistung von Funktionen

Funktionale Anforderungen vs. Nichtfunktionale Anforderungen

- Funktionale Anforderungen: Funktionen des Produkts
- Nichtfunktionale Anforderungen: "Meta"-Eigenschaften des Produkts

Zielbestimmung vs. Produkteinsatz

Zielbestimmung vs. Funktionale Anforderungen

- Zielbestimmung: allgemeine Beschreibung, was das Produkt können soll
- Funktionale Anforderungen: konkrete Auflistung von Funktionen

Funktionale Anforderungen vs. Nichtfunktionale Anforderungen

- Funktionale Anforderungen: Funktionen des Produkts
- Nichtfunktionale Anforderungen: "Meta"-Eigenschaften des Produkts

Zielbestimmung vs. Produkteinsatz

- Zielbestimmung: allgemeine Beschreibung, was das Produkt können soll
- Produkteinsatz: Rahmenbedingungen (Zielgruppe, Anwendungsbereiche)

Grundlegende Aufgabe

Erweiterung des Lastenheftes, sodass exakt abgebildet ist **was** (noch nicht **wie**) zu implementieren ist. Vom Entwickler geschrieben.

Grundlegende Aufgabe

Erweiterung des Lastenheftes, sodass exakt abgebildet ist **was** (noch nicht **wie**) zu implementieren ist. Vom Entwickler geschrieben.

- man kann es sich so merken
 - erst werden uns „Lasten“ vom Kunden auferlegt
 - daraus generieren wir dann „Pflichten“
 - „Pflichten“ Grundlage für Entwurf

- ① Zielbestimmung
- ② Produkteinsatz
- ③ **Produktumgebung** (Hard-/Software in Einsatzumgebung)
- ④ Funktionale Anforderungen
- ⑤ Produktdaten
- ⑥ Nichtfunktionale Anforderungen
- ⑦ **Globale Testfälle** („zu testende Abläufe“)
- ⑧ Systemmodelle
 - Szenarien
 - Anwendungsfälle
 - **Objektmodelle** \implies UML-Klassendiagramme (heute)
 - **Dynamische Modelle** \implies nächstes Mal
 - **Benutzerschnittstelle** \implies Zeichnungen/Screenshots
- ⑨ Glossar

Produkteinsatz vs. Produktumgebung

Produkteinsatz vs. Produktumgebung

- Produkteinsatz: Rahmenbedingungen (Zielgruppe, Anwendungsbereiche)
- Produktumgebung: Rahmenbedingungen bzgl. Software/Hardware

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes.

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase.

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr
- Nicht-funktionale Eigenschaften beschreiben, was das Produkt nicht tun sollte.

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr
- Nicht-funktionale Eigenschaften beschreiben, was das Produkt nicht tun sollte. falsch

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr
- Nicht-funktionale Eigenschaften beschreiben, was das Produkt nicht tun sollte. falsch
- Das Pflichtenheft beschreibt nur, was zu implementieren ist und nicht wie.

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr
- Nicht-funktionale Eigenschaften beschreiben, was das Produkt nicht tun sollte. falsch
- Das Pflichtenheft beschreibt nur, was zu implementieren ist und nicht wie. wahr

Wahr oder falsch?

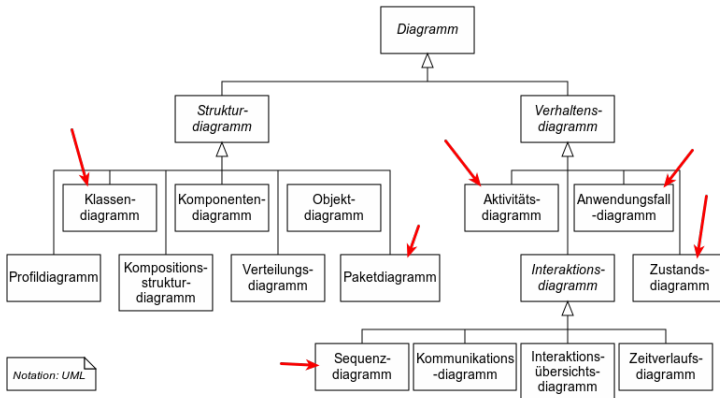
- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr
- Nicht-funktionale Eigenschaften beschreiben, was das Produkt nicht tun sollte. falsch
- Das Pflichtenheft beschreibt nur, was zu implementieren ist und nicht wie. wahr
- Nicht-funktionale Anforderungen sind sowohl Teil des Pflichtenhefts als auch des Lastenhefts.

Wahr oder falsch?

- Das Lastenheft ist eine Verfeinerung des Pflichtenheftes. falsch
- Das Lastenheft ist das Ergebnis der Planungsphase. wahr
- Nicht-funktionale Eigenschaften beschreiben, was das Produkt nicht tun sollte. falsch
- Das Pflichtenheft beschreibt nur, was zu implementieren ist und nicht wie. wahr
- Nicht-funktionale Anforderungen sind sowohl Teil des Pflichtenhefts als auch des Lastenhefts. wahr

UML? Kann man das essen?

- UML = **U**nified **M**odeling **L**anguage
- grafische Modellierungssprache, strenge Syntax



■ Klassenname

- keine spezielle Syntax
- einfach Namen hinschreiben

■ Attribute

- `<modifier><name>:<type>`

■ Methoden

- `<modifier><name>(<parameters>):<type>`
 - `<parameters>`
 - kann leer sein
 - oder komma-getrennte Liste von `<name>:<type>`
 - falls Rückgabe void, `:<type>` weglassen
- statische Methoden und Attribute unterstreichen

Name der Klasse
Attribute
Methoden

■ UML

■ -

- private
- von Instanzen derselben Klasse sichtbar (**aber von allen!**)

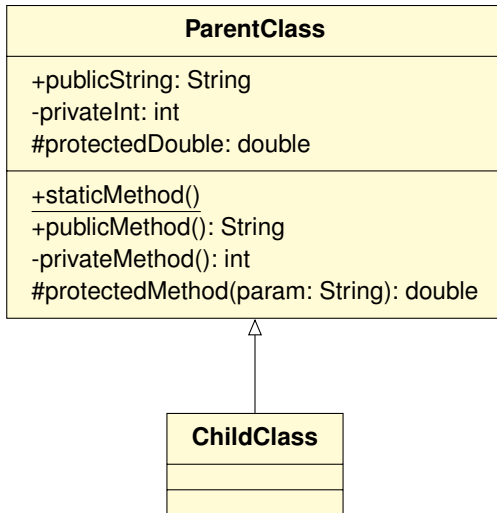
■

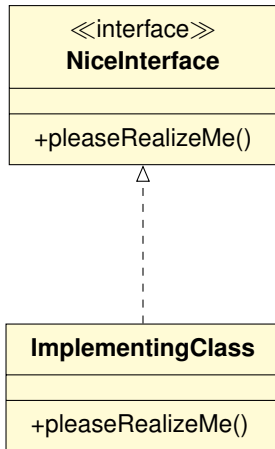
- protected (wie in Java)
- von Instanzen derselben Klasse, aller Unterklassen und Instanzen aus dem gleichen Paket sichtbar

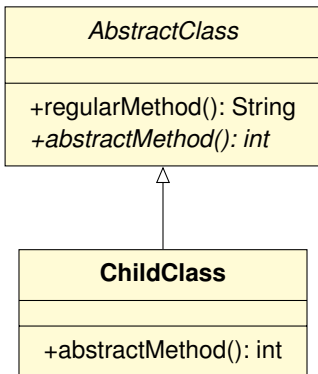
■ +

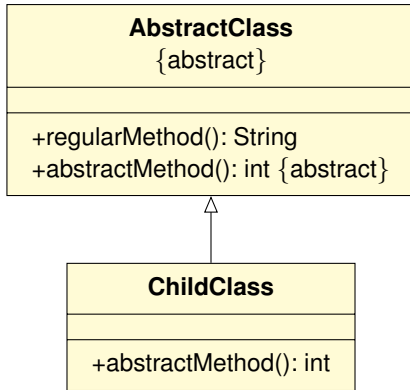
- public (wie in Java)
 - von Instanzen jeder Klasse sichtbar
- falls nichts angegeben implizit public

Hund
<ul style="list-style-type: none">- name: String- rasse: Rasse- gewicht: int
<ul style="list-style-type: none">+ wiegen(): int+ streicheln()+ streicheln(intensität: int, ausruf: String)+ füttern(ration: Nahrung)

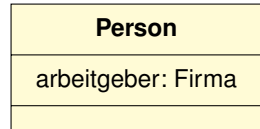
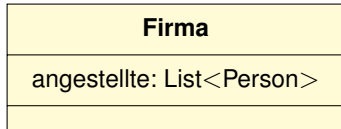








- für Übungsblätter und Klausur
 - kursiv nicht erkennbar, stattdessen **{abstract}** verwenden
 - laut VL unter Klassenname, hinter Methode



Firma
angestellte: List<Person>

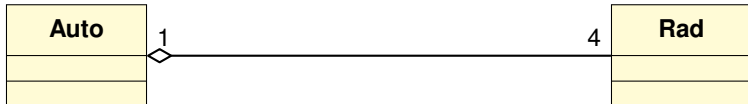
Person
arbeitgeber: Firma

Probleme

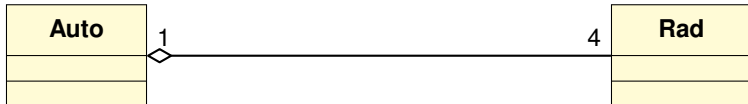
- List<X> ist Java-Syntax und schreibt Datenstruktur vor
- Beziehungen sollen direkt ersichtlich werden
- Faustregel: nur primitive Typen als Attribute hinschreiben



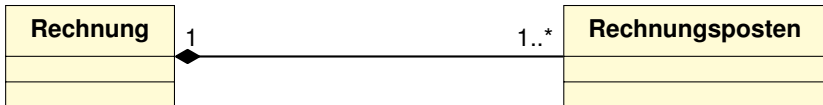
- Aggregation = Teil-Ganzes-Beziehung



- Aggregation = Teil-Ganzes-Beziehung



- Komposition: Aggregation, aber Teil kann ohne Ganzes nicht existieren
 - wenn ganzes gelöscht wird, dann auch Teile!



Text \implies UML-Klassendiagramm

Jeder Student hat eine Matrikelnummer und einen Namen. Ein fauler Student ist ein Student, der schlafen kann. Er hat dazu ein Bett. Ein fleißiger Student hingegen, kann lernen und hat dazu einen Computer, der aus Bauteilen besteht.

Text \implies UML-Klassendiagramm

Jeder Student hat eine Matrikelnummer und einen Namen. Ein fauler Student ist ein Student, der schlafen kann. Er hat dazu ein Bett. Ein fleißiger Student hingegen, kann lernen und hat dazu einen Computer, der aus Bauteilen besteht.

UML-Diagramm?

Text \Rightarrow UML-Klassendiagramm

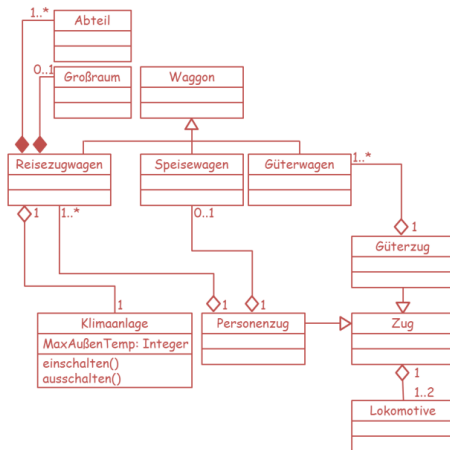
Jeder **Student** hat eine Matrikelnummer und einen Namen. Ein fauler Student **ist ein** Student, der schlafen kann. Er **hat** dazu ein Bett. Ein fleißiger Student hingegen, kann lernen und **hat** dazu einen **Computer**, der aus **Bauteilen besteht**.

Schlüsselwörter!

Modellieren Sie das Szenario möglichst vollständig als UML-Klassendiagramm. Geben Sie Methoden, Attribute, Multiplizitäten, Restriktionen, Assoziationsnamen, Aggregationen und Kompositionen sowie Rollen an.

Szenario

Ein Güterzug ist ein Zug, dessen Waggon ausschließlich Güterwagen sind. Die Waggon eines Personenzugs sind mindestens ein Reisezugwagen und höchstens ein Speisewagen. Jeder Zug hat eine oder zwei Lokomotiven. Reisezugwagen setzen sich aus bis zu einem Großraum sowie einem oder mehreren Abteilen zusammen. Jeder Reisezugwagen hat eine Klimaanlage, die ein- und ausgeschaltet werden kann. Jede Klimaanlage darf nur bis zu einer bestimmten maximalen Außentemperatur betrieben werden.

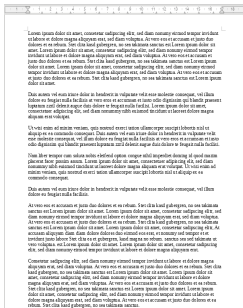


Klassen	11x	0,5 P
Methoden/Attribute "Klimaanlage"	1x	0,5 P
Aggregationen/Kompositionen	7x	0,5 P
Vererbung (alle 3 zusammen)	1x	0,5 P

- auf dem Blatt müsst ihr L^AT_EX für die Dokumente benutzen
- wird euch an der Uni immer wieder begegnen, manchmal Pflicht

- auf dem Blatt müsst ihr L^AT_EX für die Dokumente benutzen
- wird euch an der Uni immer wieder begegnen, manchmal Pflicht
- nicht WYSIWYG
 - What You See Is What You Get
 - Paradigma von Word etc.

(a) What You See



(b) What You Get



- stattdessen WYSIWYAF bzw. WYSIWYM
 - What You See Is What You Ask For / Mean
 - Paradigma von L^AT_EX \approx HTML und CSS

(a) What You See

```


\subsection{Pflichtenheft - Gliederung}
\begin{frame}{Pflichtenheft - Gliederung}
\begin{enumerate}
\item Zielbestimmung
\item Produkteinsatz
\item \underline{\textbf{Produktumgebung}} (Hard-/Software in \textit{Einsatzumgebung})
\item Funktionale Anforderungen
\item Produktdaten
\item Nichtfunktionale Anforderungen
\item \underline{\textbf{Globale Testfälle}} (\enquote{zu testende Abläufe})
\item Systemmodelle
\begin{itemize}
\item Szenarien
\item Anwendungsfälle
\item \underline{\textbf{Objektmodelle}} $\implies$ UML-Klassendiagramme (heute)
\item \underline{\textbf{Dynamische Modelle}} $\implies$ nächstes Mal
\item \underline{\textbf{Benutzerschnittstelle}} $\implies$ Zeichnungen/Screenshots
\end{itemize}
\item Glossar
\end{enumerate}
\end{frame}

```

(b) What You Mean

Pflichtenheft - Gliederung

- 1 Zielbestimmung
- 2 Produkteinsatz
- 3 **Produktumgebung** (Hard-/Software in Einsatzumgebung)
- 4 Funktionale Anforderungen
- 5 Produktdaten
- 6 Nichtfunktionale Anforderungen
- 7 **Globale Testfälle** („zu testende Abläufe“)
- 8 Systemmodelle
 - Szenarien
 - Anwendungsfälle
 - **Objektmodelle** \implies UML-Klassendiagramme (heute)
 - **Dynamische Modelle** \implies nächstes Mal
 - **Benutzerschnittstelle** \implies Zeichnungen/Screenshots
- 9 Glossar



Orga	Wasserfallmodell	Durchführbarkeitsuntersuchung	Lastenheft	Pflichtenheft	UML-Klassendiagramm	ITgX	Tipp
000000	000	0	00	0000	00000000000000	0000	000
Felix Bachmann – SWT1					14.05.2019	15/38	

■ Vorteile

- gut versionierbar
 - „Quellcode“ ist normales Textdokument
 - Word etc. verwenden oft XML-Formate mit Metadaten
- leicht Formeln erstellbar
- nach Eingewöhnung recht intuitiv
- multifunktional (Bücher, Dokumente, Präsentationen, ...)
- kostet nix :)
- open source
- viele Erweiterungen, Pakete, ...

■ Nachteile

- Einarbeitung notwendig :(

Installation einer Distribution notwendig, z.B.:

- MikTeX für Windows
- TeX Live für Linux, Mac, Windows

Installation einer Distribution notwendig, z.B.:

- MikTeX für Windows
- TeX Live für Linux, Mac, Windows

Editoren machen das Schreiben von L^AT_EX-Dokumenten angenehmer

- Texmaker
- TeXstudio (erweiterter Texmaker, mein Favorit)
- TeXclipse (Plugin für Eclipse)
- Plugins für Visual Studio Code
- ...

Präambel: Pakete laden, Dokumenttyp festlegen

```
\documentclass{Klasse}  
\usepackage[option1,option2,...]{Paket}
```

- nützliche Klassen: book, beamer, scrartcl
- nützliches Paket z.B. csquotes (ermöglicht `\enquote{...}`)

Präambel: Pakete laden, Dokumenttyp festlegen

```
\documentclass{Klasse}  
\usepackage[option1,option2,...]{Paket}
```

- nützliche Klassen: book, beamer, scrartcl
- nützliches Paket z.B. csquotes (ermöglicht `\enquote{...}`)

Inhalt: Text setzen, Bilder, Graphiken, Formeln,...

```
% preamble  
\begin{document}  
    content  
\end{document}
```

- Struktur: part, (chapter), section, subsection, subsubsection
- Auflistungen: `\begin{itemize}` `\item Hello World!` `\end{itemize}`
- Bilder: `\includegraphics[scale = 0.8]{PfadZumBild}`

L^AT_EX-Beispiel!

Aufgabe 1 + 3: Lastenheft + Durchführbarkeitsuntersuchung

- lasst euch was (sinnvolles) einfallen
- benutzt \LaTeX

Aufgabe 1 + 3: Lastenheft + Durchführbarkeitsuntersuchung

- lasst euch was (sinnvolles) einfallen
- benutzt \LaTeX

Aufgabe 2: Klassendiagramme

- achtet auf Schlüsselwörter (“ist ein“, “enthält ein“, “besteht aus“, ...)

Aufgabe 1 + 3: Lastenheft + Durchführbarkeitsuntersuchung

- lasst euch was (sinnvolles) einfallen
- benutzt \LaTeX

Aufgabe 2: Klassendiagramme

- achtet auf Schlüsselwörter (“ist ein“, “enthält ein“, “besteht aus“, ...)

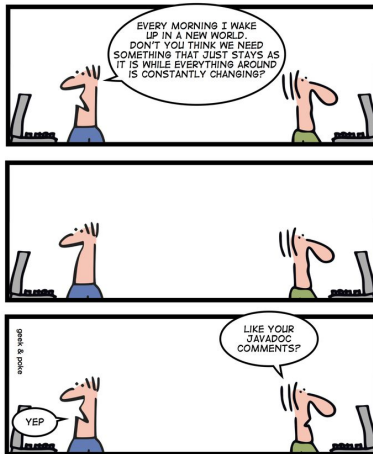
Aufgabe 4 + 5: Shutterpile

- an einigen Stellen sind Aufgaben etwas vage
 \implies überlegt euch, was Sinn macht
- Zusammenhang der Klassen unklar? Vielleicht hilft Diagramm

Abgabe

- Deadline am 16.5 um 12:00
- Dokumente ausdrucken
- Klassendiagramme handschriftlich

Bis dann! (dann := 22.05.18)



DEVELOPERS