



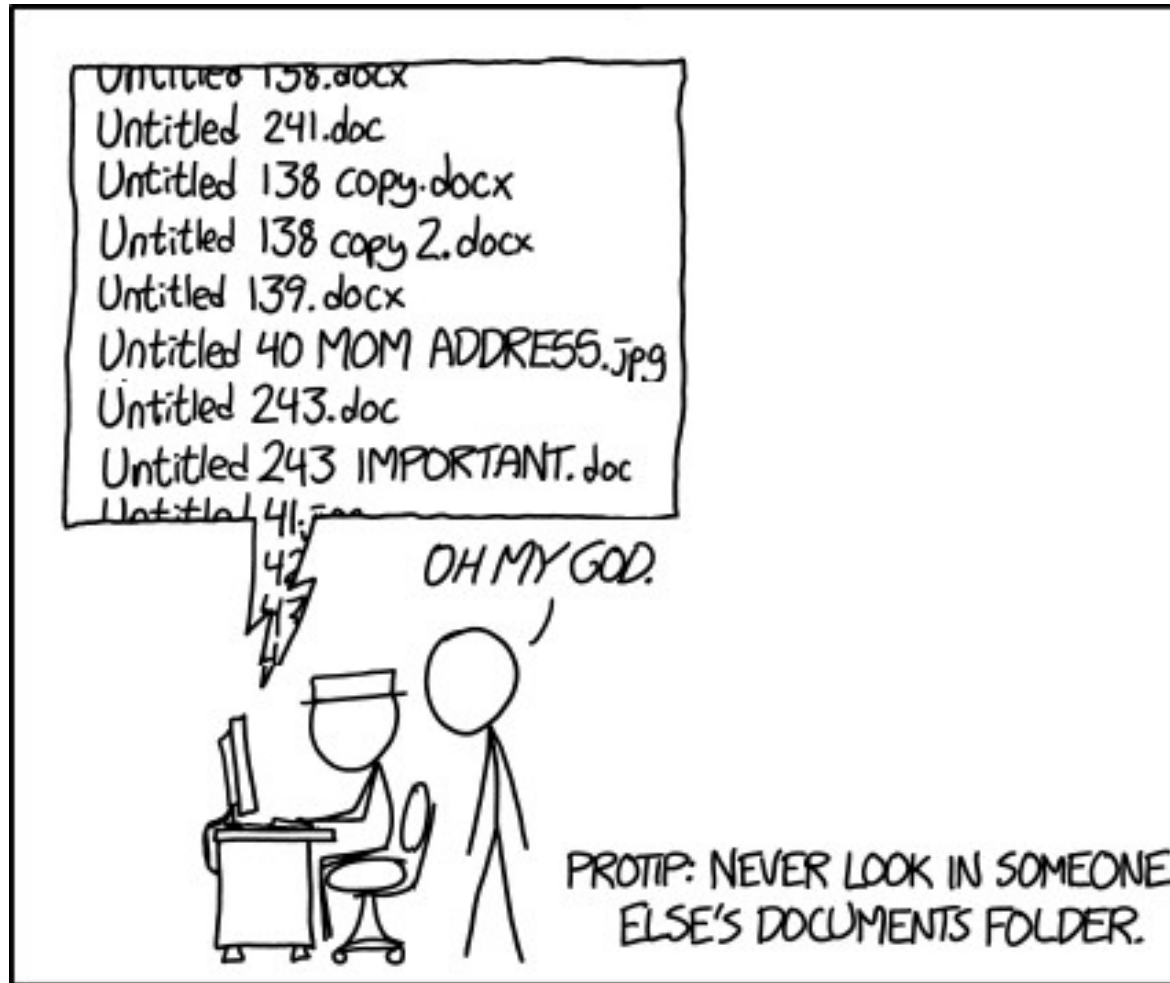
Thomas Rau  
(mit Peter Brichzin)

# BlueJ mit Repositories (Subversion)

# In Softwareprojekten gibt es oft organisatorische Probleme, die von der inhaltlichen Arbeit ablenken

- Wie lassen sich die Teilergebnisse regelmäßig zusammenfügen?
- Wie vermeidet man, dass das Team blockiert ist, weil der Schüler mit wesentlichen Teilen des aktuellen Quelltextes krank zu Hause ist?
- Wie ermöglicht man, dass begeisterte Schülerinnen und Schüler zu Hause weiterentwickeln können?
- Wie lässt sich einfach dokumentieren, wer wann welche Teile der Software bearbeitet hat?

# Dateien können in unterschiedlicher Qualität verwaltet werden



Quelle: <https://xkcd.com/1459/>

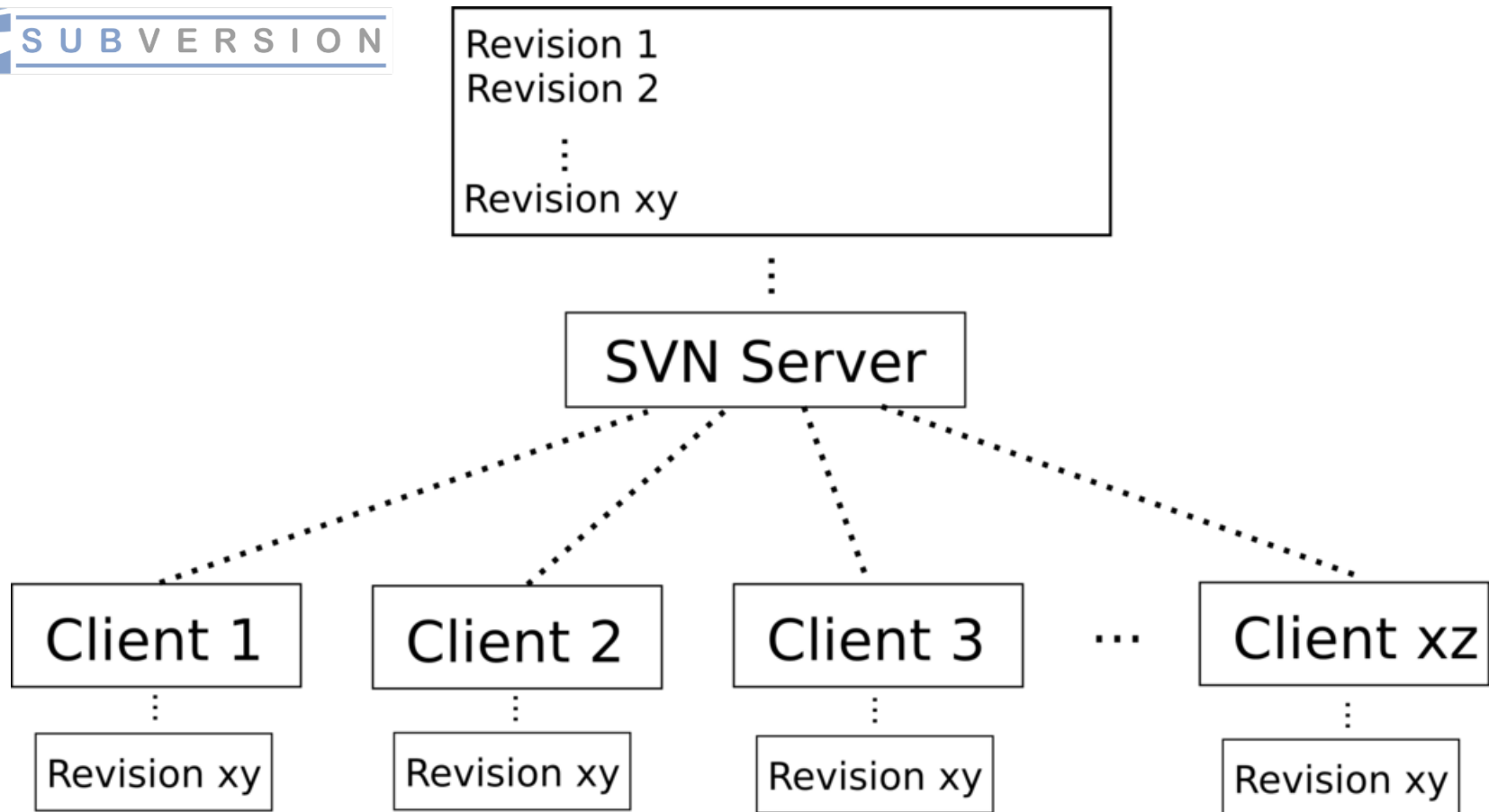
# Versionskontroll-Systeme lösen die dargestellten Probleme

- **Verteilter Zugriff** auf Dateien als Voraussetzung für verteiltes kollaboratives Arbeiten
- **Versionierung**: Übersicht über verschiedene Versionen inklusiv der **Dokumentation** wer, wann, was geändert hat
- **Datensicherheit**, u.a. durch das Rücksetzen der Datei auf eine ältere Version
- **Automatisches Zusammenführen** (engl. merge) von Quelltexten bzw. Konflikterkennung, falls mehrere Entwickler Veränderungen im selben Bereich durchführen

# Überblick

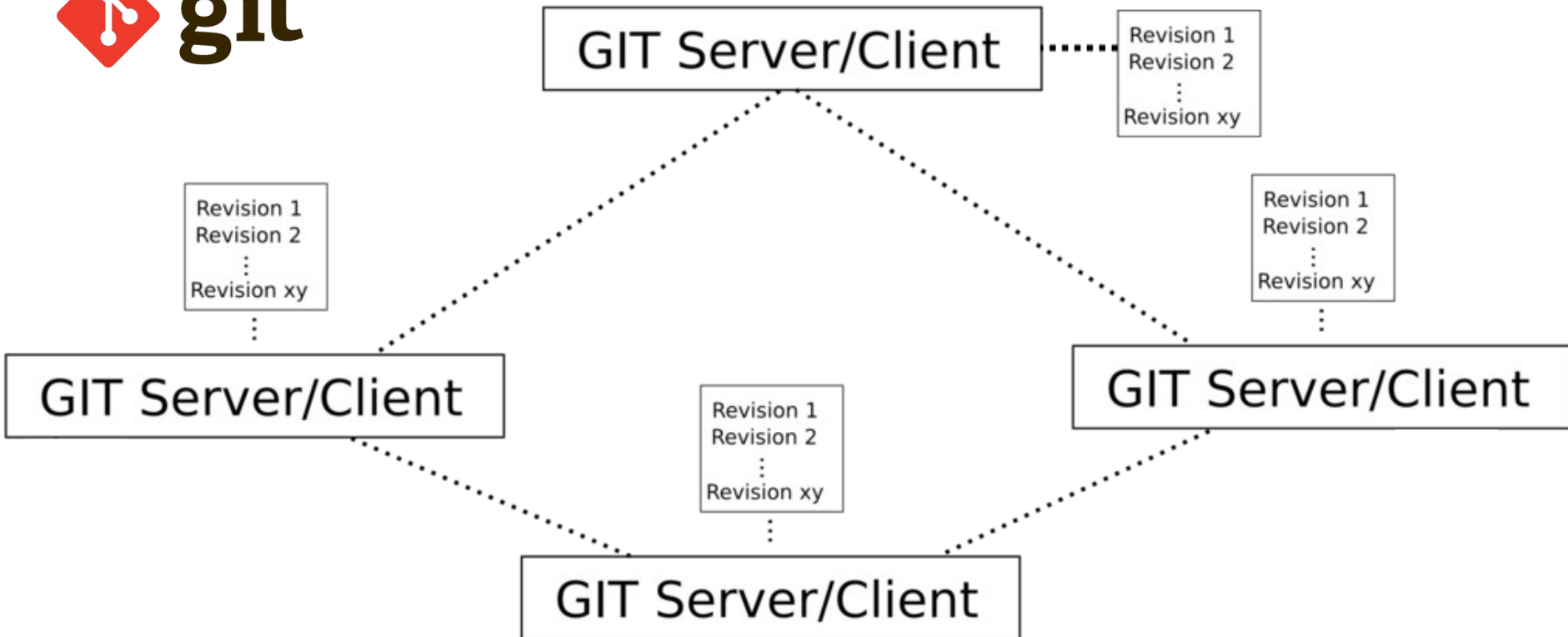
1. **Grundlegendes zu Versionsverwaltung**
2. **Aufgabe 1, Teil 1: Auschecken**
3. **Aufgabe 1, Teil 2: Abgeben und Aktualisieren**
4. **Aufgabe 2: Konflikte**
5. **Aufgabe 3: Projekt erstellen**
6. **SVN-Server**
7. **Workflow für Lehrer**
8. **Erfahrungen**
9. **Anwendungsbeispiel Entwurfsmuster MVC**

# Bei einer zentralen Versionsverwaltung werden bei Änderungen nur die Unterschiede übertragen



[https://commons.wikimedia.org/wiki/File:SVNvsGITServer\\_1.png](https://commons.wikimedia.org/wiki/File:SVNvsGITServer_1.png)  
Paul Vincent, Creative Commons Attribution 3.0 Unported

# Bei einem verteilten Versionsverwaltungssystem besitzt jeder eine lokale Kopie inkl. der Historie



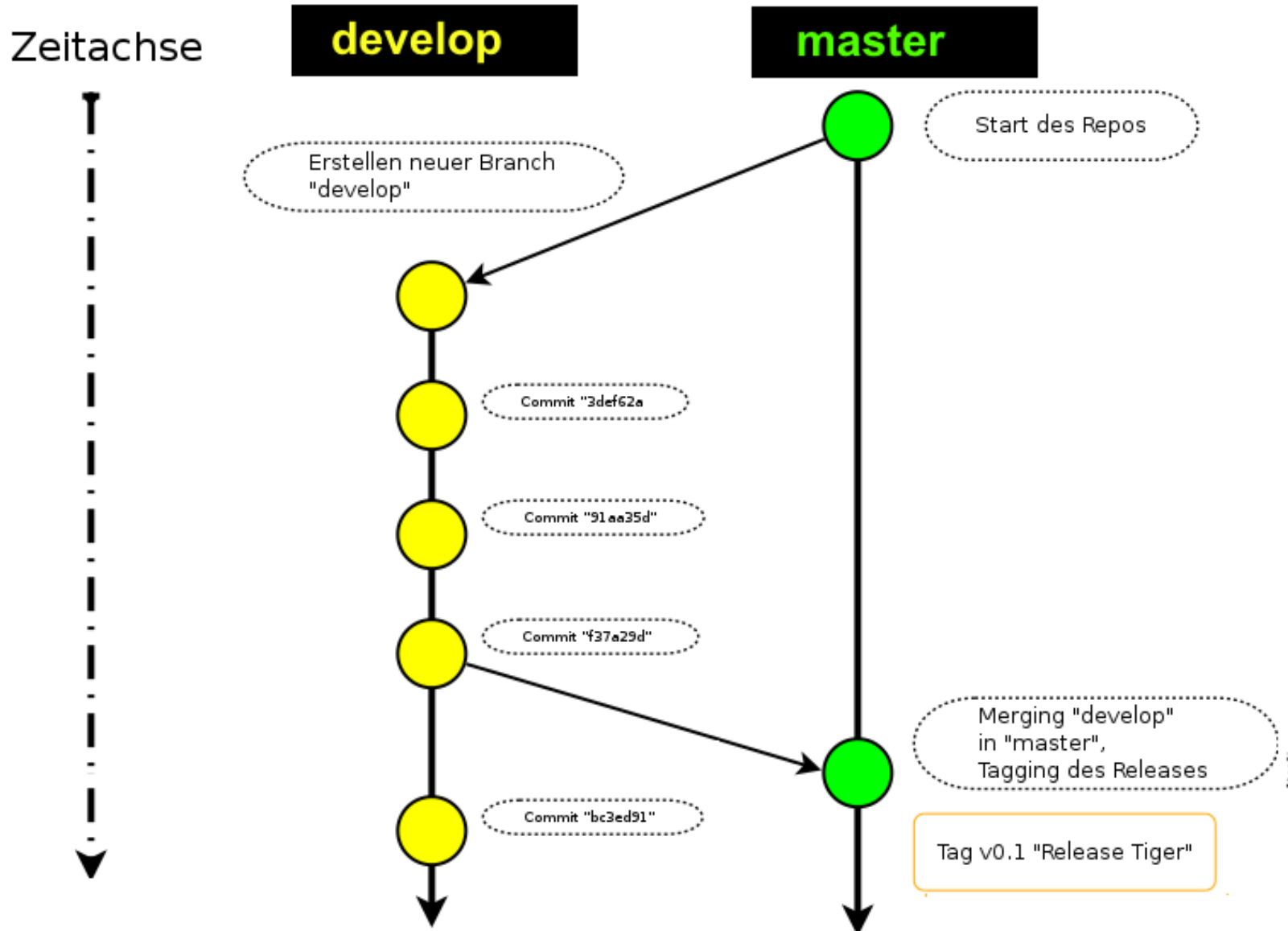
Quelle: [https://commons.wikimedia.org/wiki/File:SVNvsGITServer\\_2.png](https://commons.wikimedia.org/wiki/File:SVNvsGITServer_2.png)  
 Paul Vincent, Creative Commons Attribution 3.0 Unported

# Die grundlegenden Operationen bei Repositories sind Checkout, Commit und Update

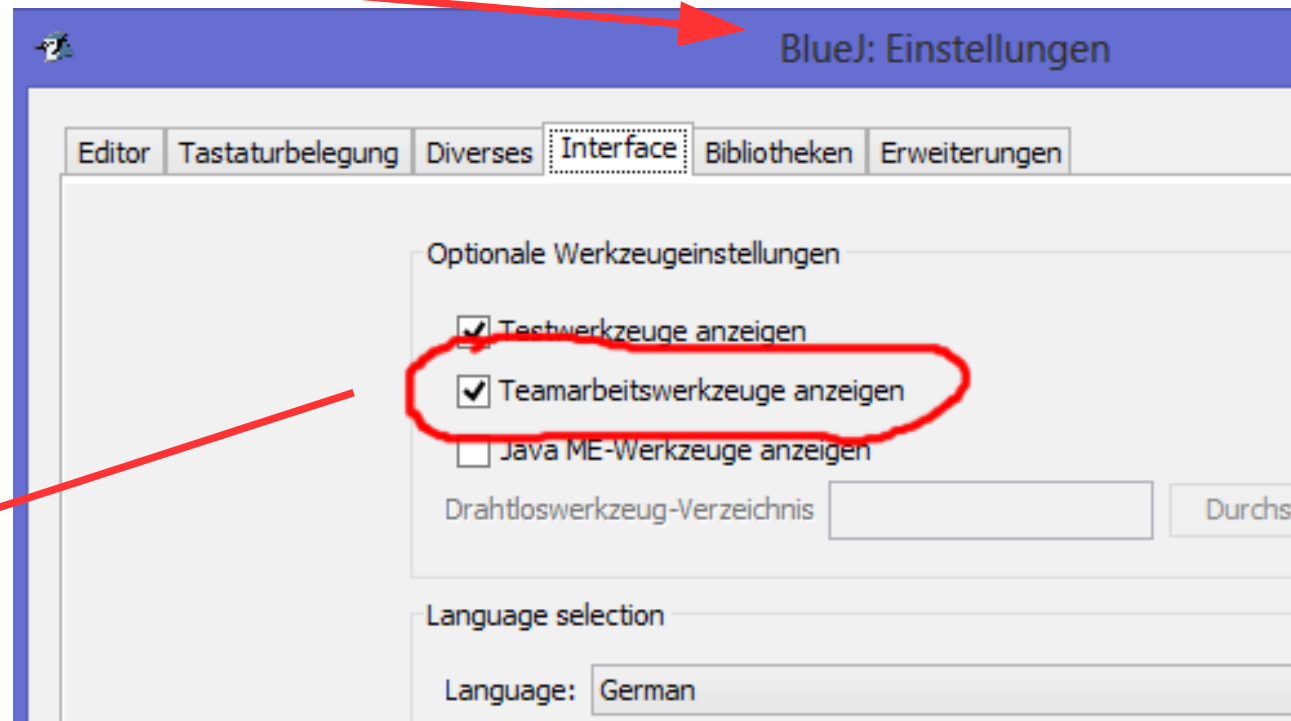
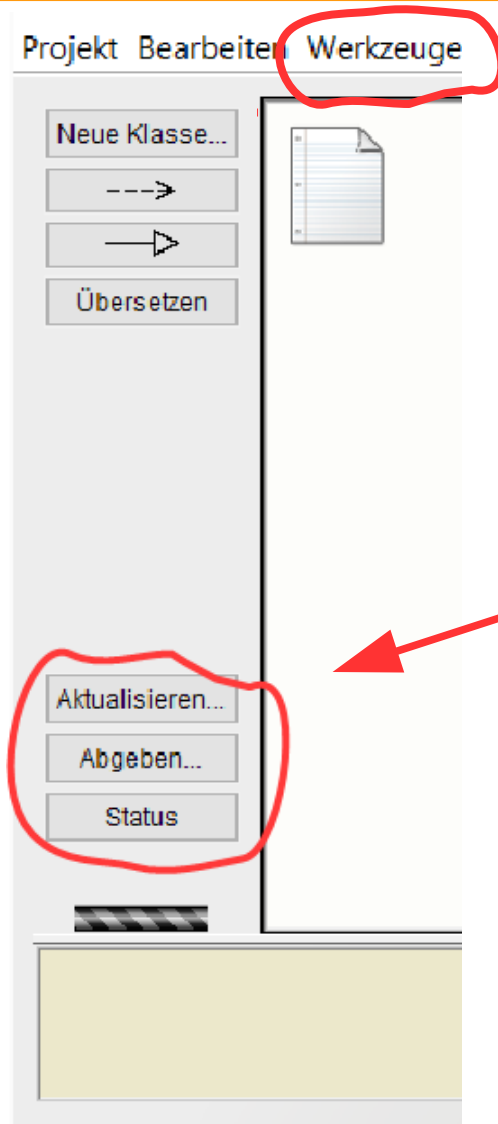
- **Checkout – *Arbeitskopie erstellen:***  
Lokal alle Inhalte des Versionskontrollsystems neu laden.
- **Commit – *Abgeben:***  
Eine Änderung am Inhalt des Versionskontrollsystems: Eine semantische Einheit ohne Übersetzungsfehler.
- **Update – *Aktualisieren:***  
Von anderen abgegebene Änderungen lokal laden.



# Die Profis verwenden zwischen Entwicklung und Releases unterschiedliche Zweige.



# In der Entwicklungsumgebung BlueJ ist ein SVN-Client integriert (fehlerfrei ab Version 3.1.7)



**Teamarbeits-Menü sichtbar machen**

# Überblick

1. **Grundlegendes zu Versionsverwaltung**
2. **Aufgabe 1, Teil 1: Auschecken**
3. **Aufgabe 1, Teil 2: Abgeben und Aktualisieren**
4. **Aufgabe 2: Konflikte**
5. **Aufgabe 3: Projekt erstellen**
6. **SVN-Server**
7. **Workflow für Lehrer**
8. **Erfahrungen**
9. **Anwendungsbeispiel Entwurfsmuster MVC**

# Aufgabe 1, Teil 1: Erstmals ein BlueJ-Projekt aus einem Repository auschecken

**Menü Werkzeuge:**  
**Teamarbeit**  
→ **Arbeitskopie erstellen**

Teamarbeitseinstellungen

**Persönlich**

Benutzer: fortbildung

Passwort: ••••••••

Gruppe:

**Repository**

Server-Typ: Subversion

Server: n.code.sf.net/p/fortbildung/code

Verzeichnis:

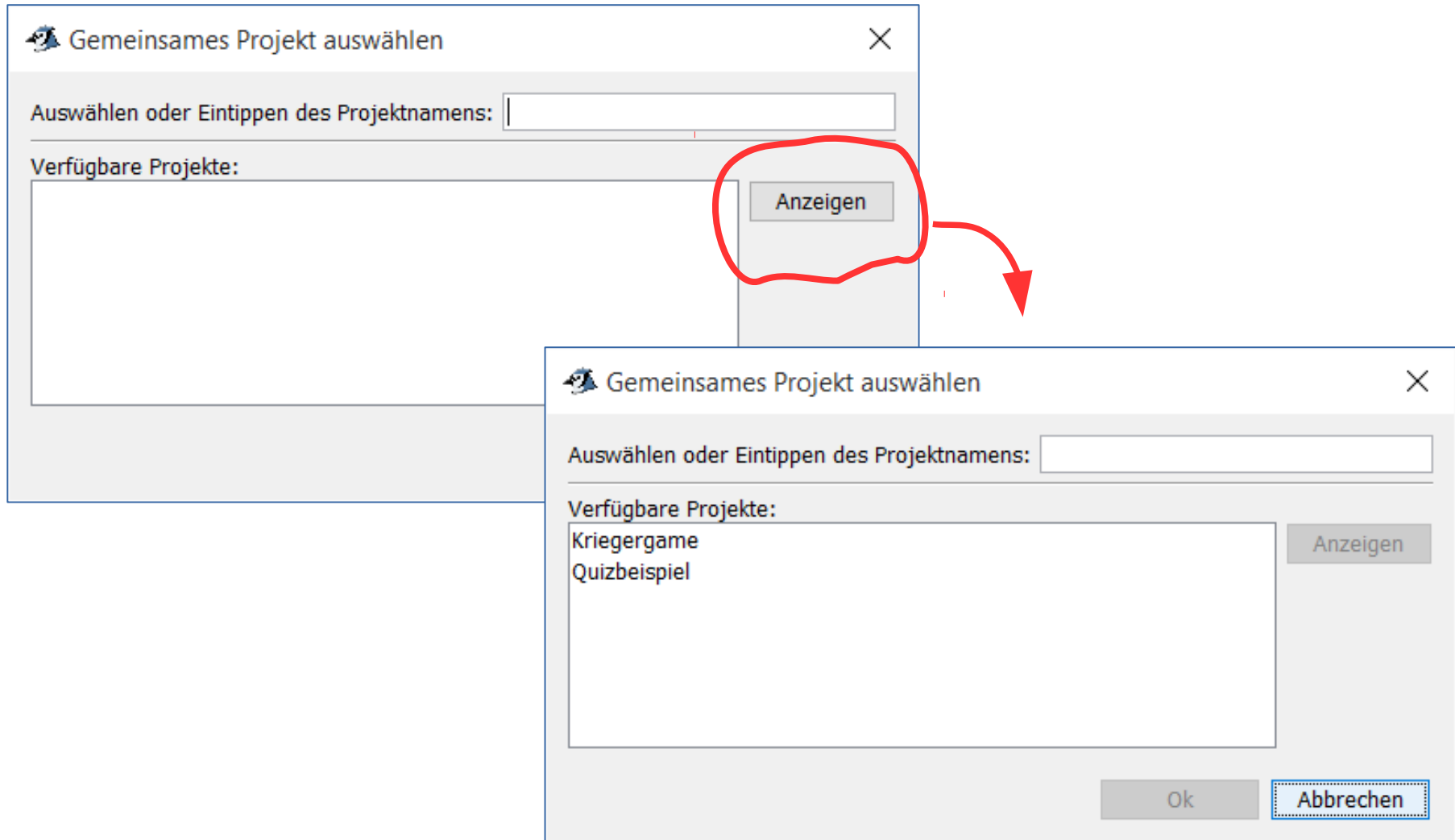
Protokoll: https

☒ Einstellungen für zukünftige Projekte speichern

Teste Verbindung

Ok Abbrechen

# Aufgabe 1, Teil 1: Erstmals ein BlueJ-Projekt aus einem Repository auschecken



# Aufgabe 1, Teil 1: Erstmalig ein BlueJ-Projekt aus einem Repository auschecken

## **Aufgabe:**

Checken Sie beliebige Projekte aus dem Server aus, aber auf jeden Fall das BlueJ-Projekt „SVN Aufgabe 1“ für den nächsten Schritt.

**Server-Typ:** Subversion

**Benutzer:** fortbildung

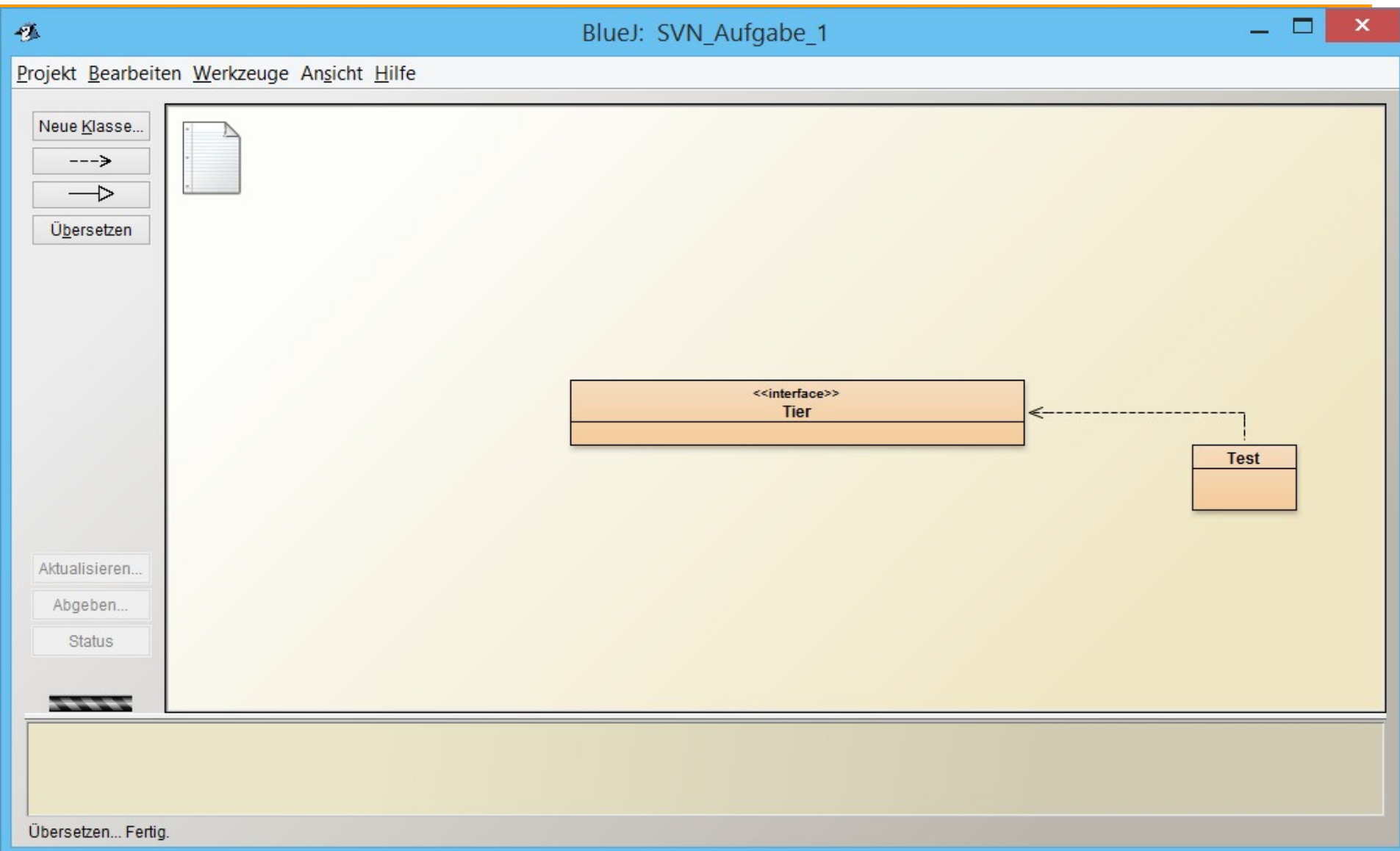
**Passwort:** [siehe Papier]

**Server:** [svn.code.sf.net/p/fortbildung/code](https://svn.code.sf.net/p/fortbildung/code)

**Path:** [leer]

**Protokoll:** https

# Aufgabe 1, Teil 1: Auschecken



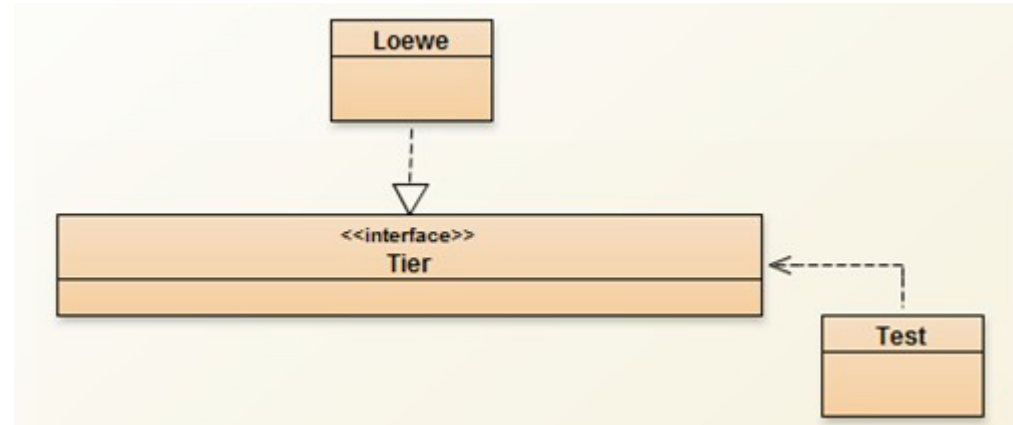
# Überblick

1. **Grundlegendes zu Versionsverwaltung**
2. **Aufgabe 1, Teil 1: Auschecken**
3. **Aufgabe 1, Teil 2: Abgeben und Aktualisieren**
4. **Aufgabe 2: Konflikte**
5. **Aufgabe 3: Projekt erstellen**
6. **SVN-Server**
7. **Workflow für Lehrer**
8. **Erfahrungen**
9. **Anwendungsbeispiel Entwurfsmuster MVC**

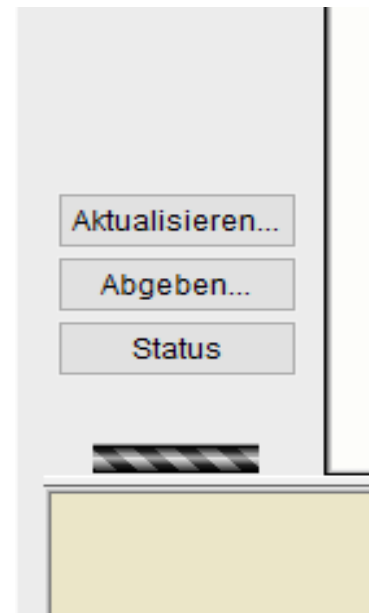


## Aufgabe 1, Teil 2: Abgeben (Commit) und Aktualisieren (Update)

Legen Sie in „SVN Aufgabe 1“ eine neue Klasse an, die das Interface „Tier“ implementiert. Die Klasse „Loewe“ kann als Beispiel dienen.

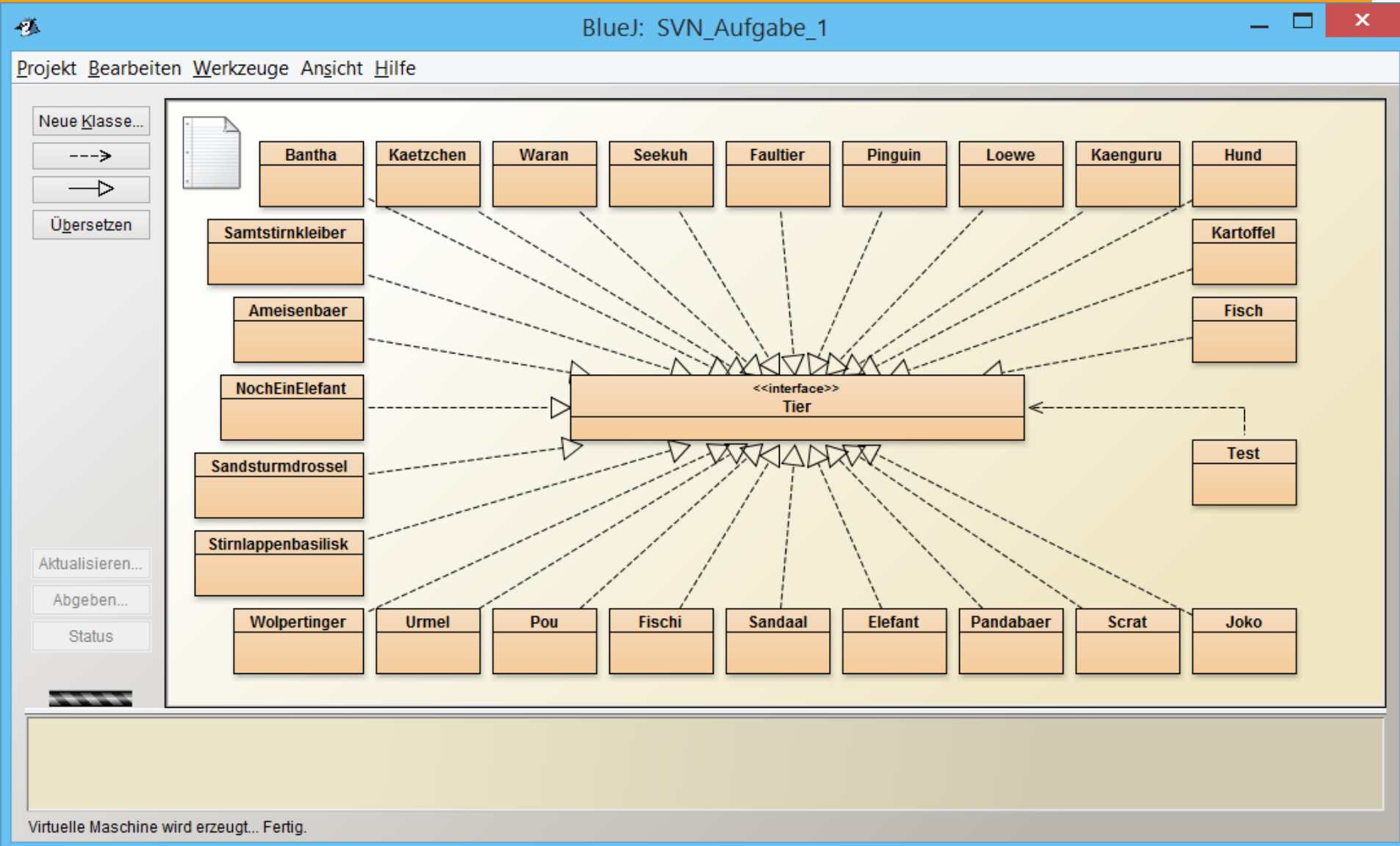


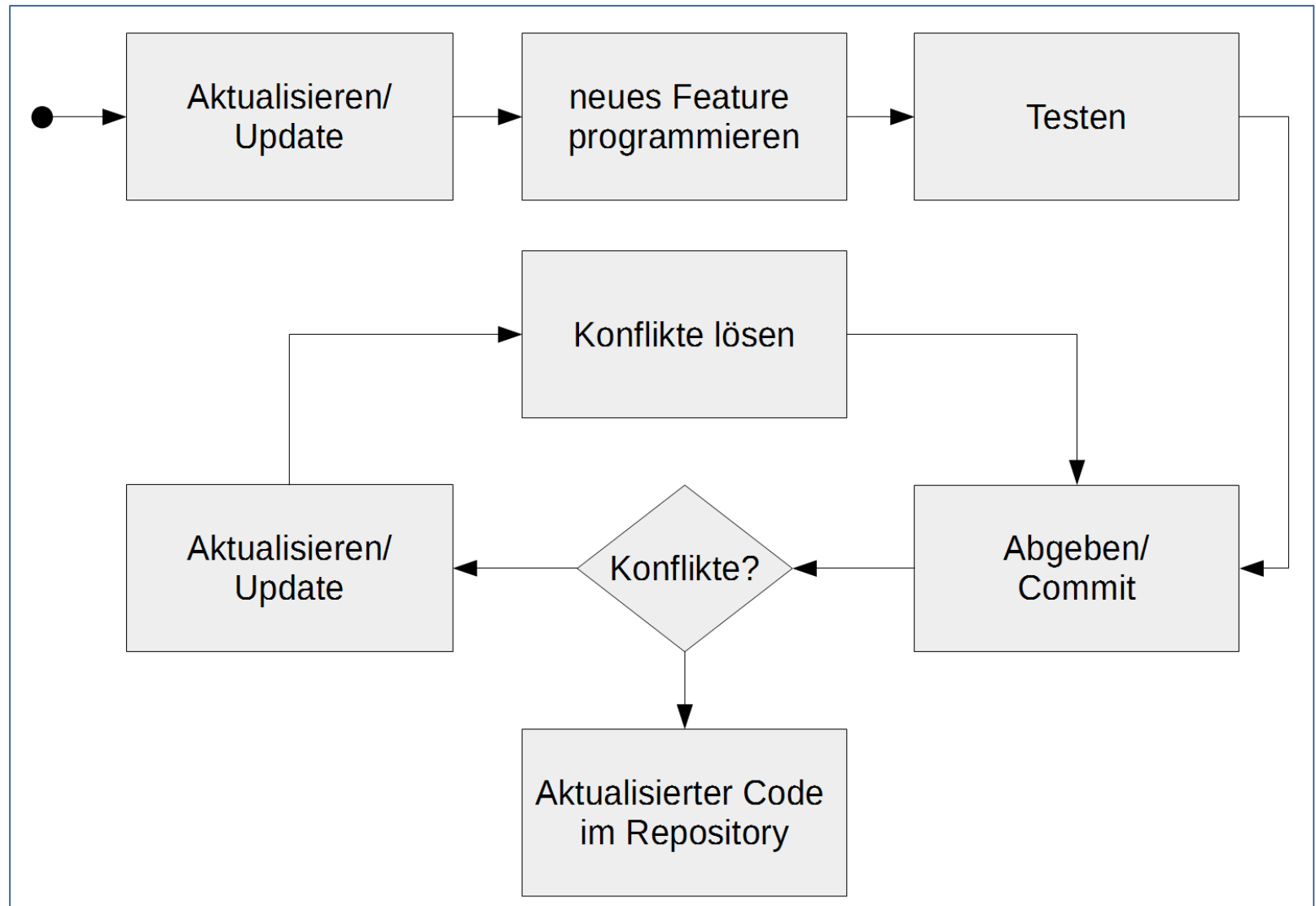
- Aktualisieren Sie.
- Geben Sie Ihre Änderungen mit einer aussagekräftigen Beschreibung (Kommentar) ab.
- Führen Sie weitere Updates und Commits durch. Sollten irgendwelche Konflikte auftreten, so ignorieren Sie diese vorläufig.





# Aufgabe 1, Teil 2: Schülerergebnisse





# Das Repository speichert die Historie

Rufen Sie den Status ab.







Teamwork status		
Team resources	Revision	Teamwork status
(Diagram Layout)	10	Modified locally
SVN AUFGABE 1	7	Up-to-date
Tier.java	5	Up-to-date
README.TXT	5	Up-to-date
Kakapo.java	7	Up-to-date

Aktualisieren...

Abgeben...

Status

Nachteil an BlueJ-Client: Keine Information über Autor und Datum

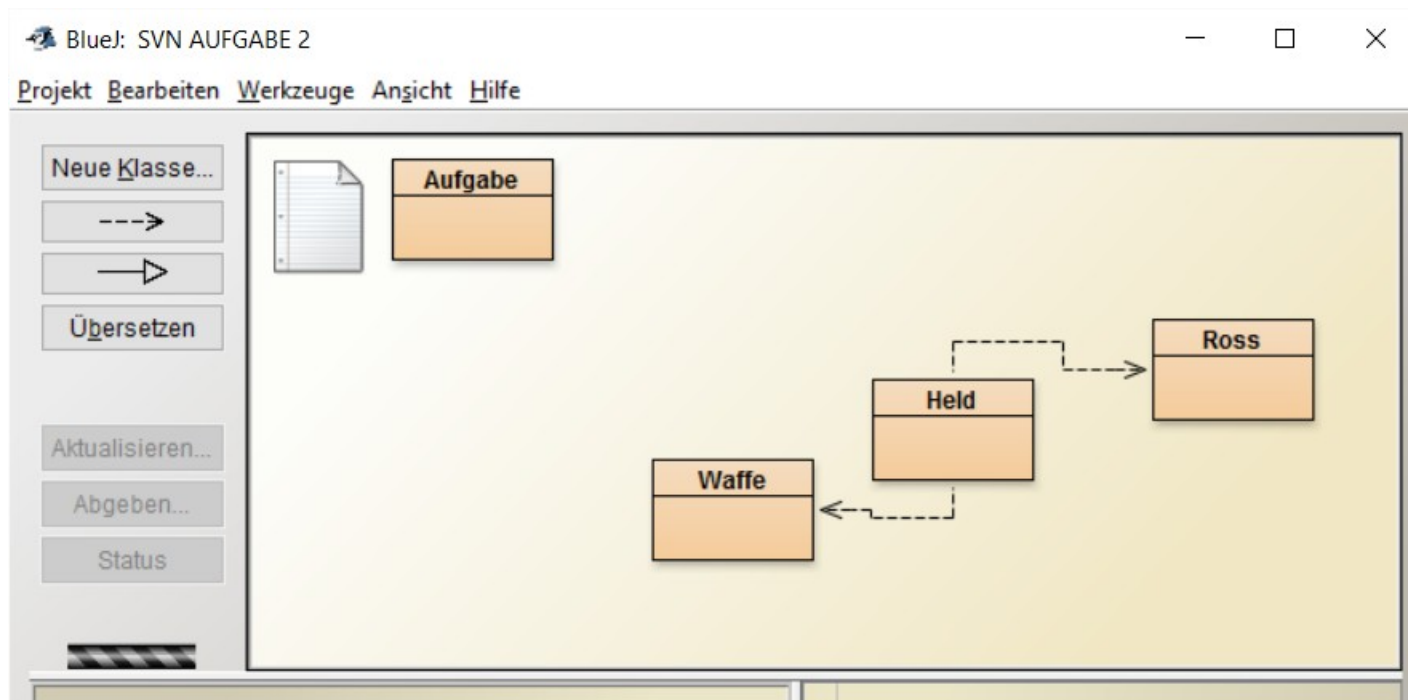
https://svn.code.sf.net/p/fortbildung/code - Repository Browser - TortoiseS... <span>✖</span>							
URL:  svn.code.sf.net/p/fortbildung/code/SVN AUFGABE 1/Fisch.java		Revision: HEAD					
File	Extension	Revision	Author	Size	Date	Lock	Lock comment
 package.bluej	.bluej	10	fortbildung	2,37 KB	17.09.2015 09:28:39		
 Fisch.java	.java	9	fortbildung	502 Bytes	17.09.2015 09:28:36		
 Kakapo.java	.java	7	herrrau	87 Bytes	26.08.2015 15:34:05		
 Kakapo.ctxt	.ctxt	7	herrrau	102 Bytes	26.08.2015 15:34:05		
 Kakapo.class	.class	7	herrrau	349 Bytes	26.08.2015 15:34:05		

# Überblick

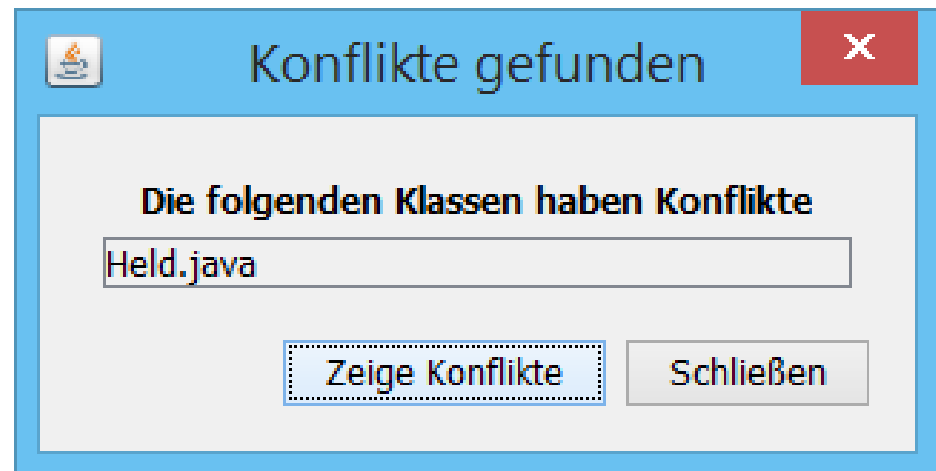
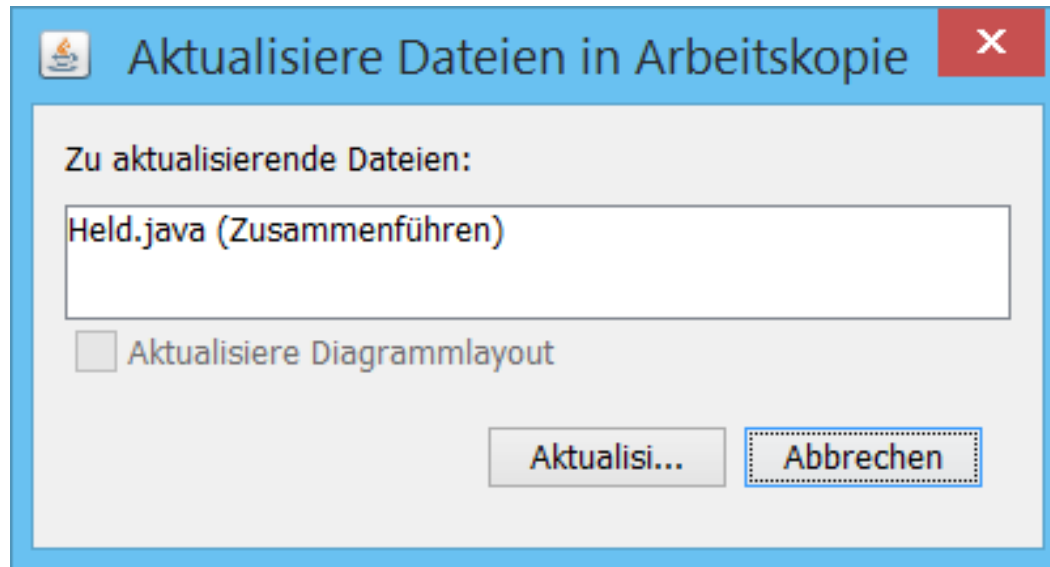
1. Grundlegendes zu Versionsverwaltung
2. Aufgabe 1, Teil 1: Auschecken
3. Aufgabe 1, Teil 2: Abgeben und Aktualisieren
4. Aufgabe 2: Konflikte
5. Aufgabe 3: Projekt erstellen
6. SVN-Server
7. Workflow für Lehrer
8. Erfahrungen
9. Anwendungsbeispiel Entwurfsmuster MVC

## Aufgabe 2: Konflikte

Aufgabe: Checken Sie das Projekt „SVN Aufgabe 2“ aus und ergänzen Sie eine noch nicht vorhandene getter- oder setter-Methode für ein Attribut der Klasse „Held“. Führen Sie dann einen Commit durch (bzw. vorher ein eventuell eingefordertes Update).

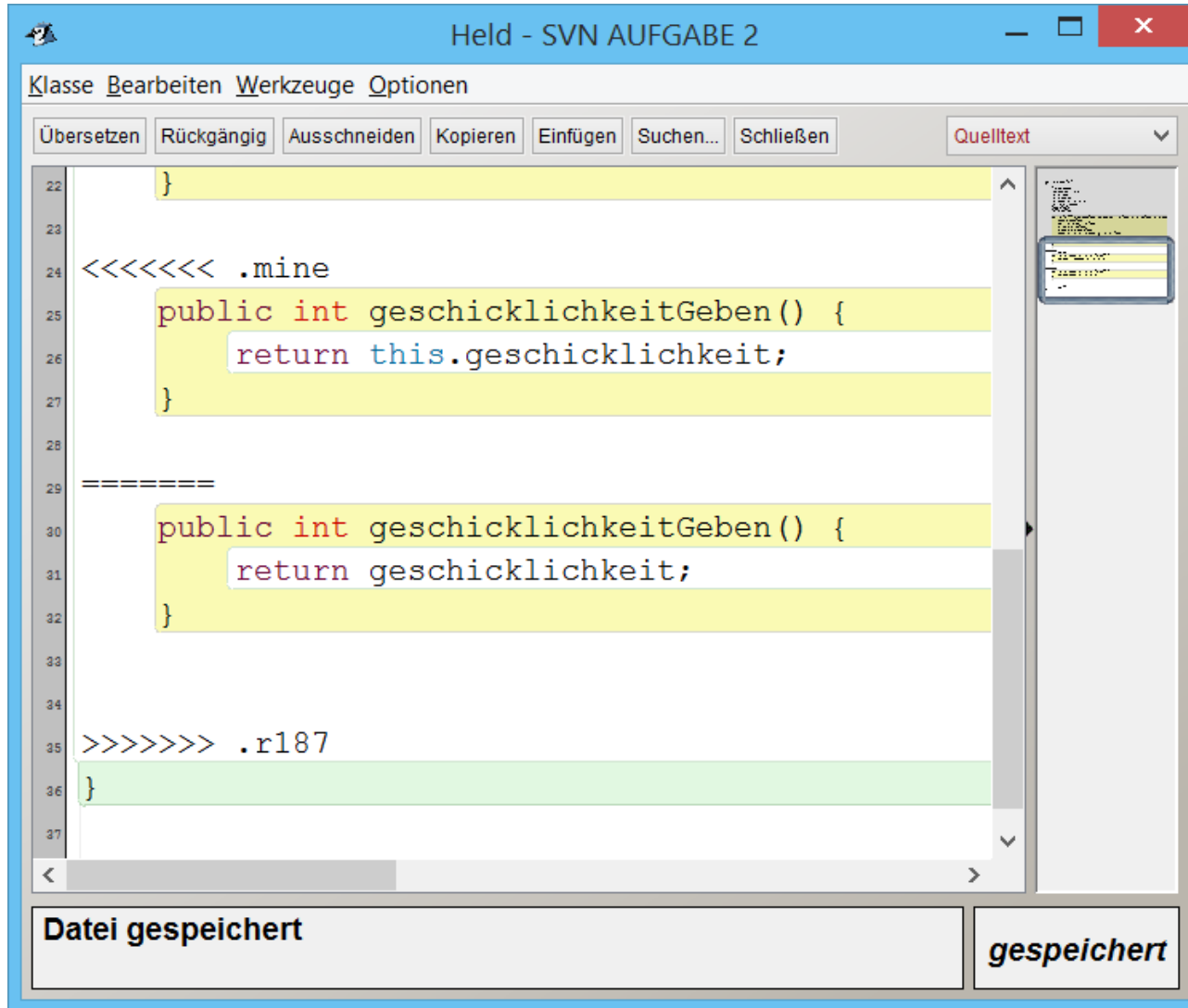


## Aufgabe 2: Konflikte automatisch zusammenführen





# Aufgabe 2: Konflikte manuell zusammenführen



```

22     }
23
24 <<<<<<< .mine
25     public int geschicklichkeitGeben() {
26         return this.geschicklichkeit;
27     }
28
29 =====
30     public int geschicklichkeitGeben() {
31         return geschicklichkeit;
32     }
33
34 >>>>>>> .r187
35 }
36
37

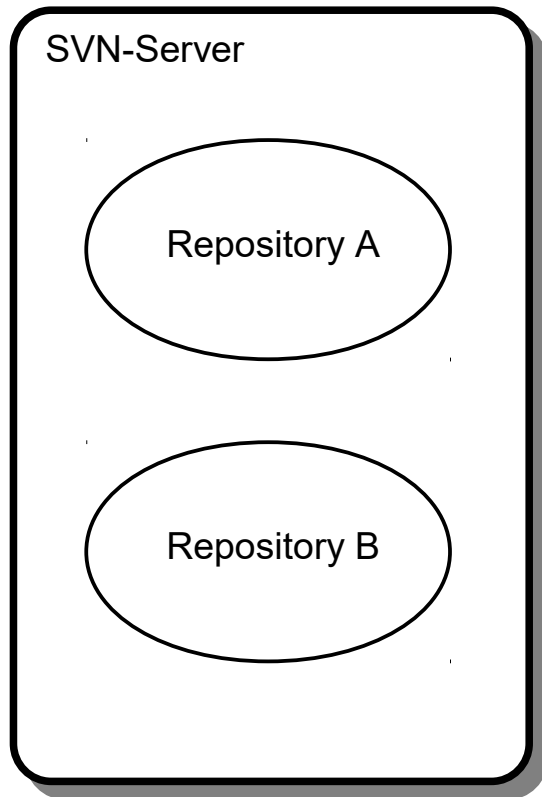
```

**Datei gespeichert** *gespeichert*

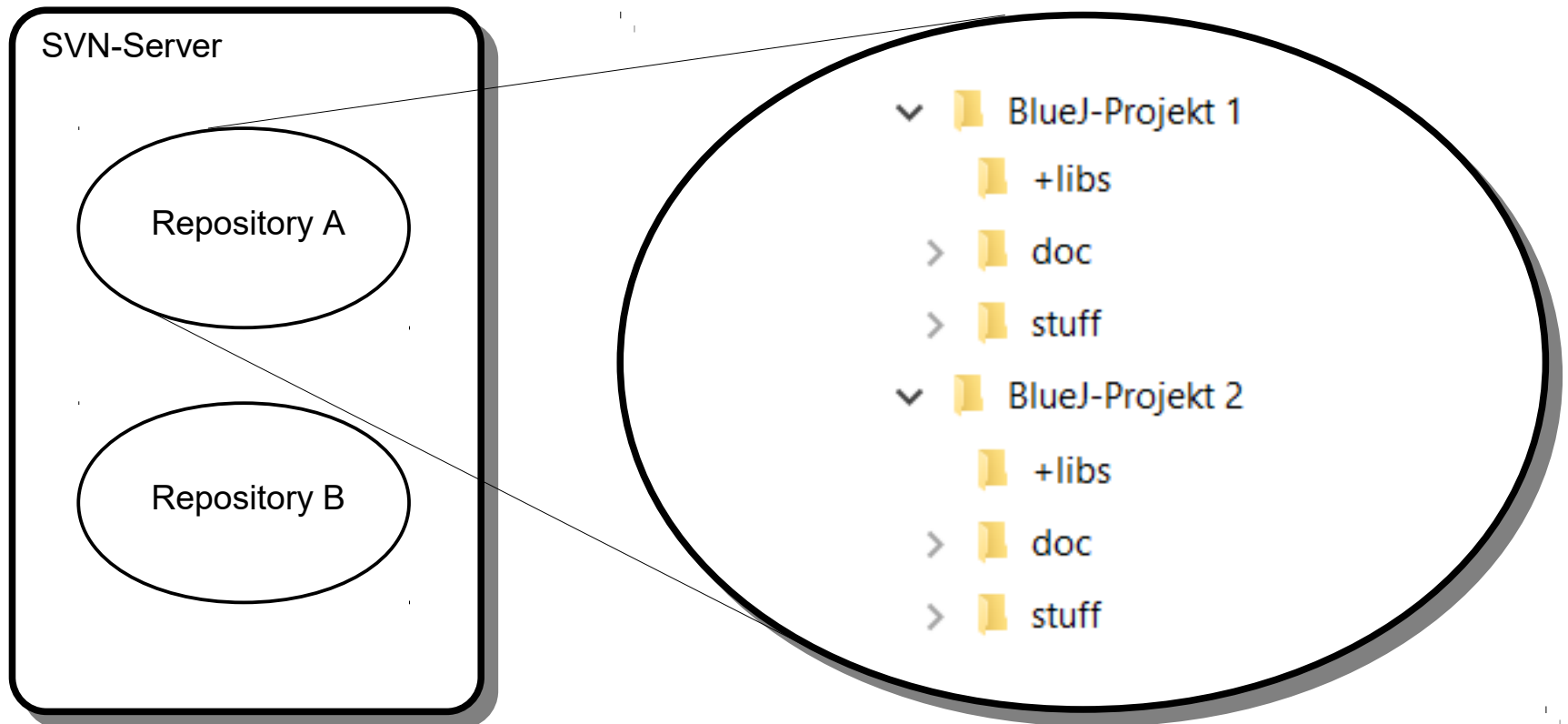
# Überblick

1. **Grundlegendes zu Versionsverwaltung**
2. **Aufgabe 1, Teil 1: Auschecken**
3. **Aufgabe 1, Teil 2: Abgeben und Aktualisieren**
4. **Aufgabe 2: Konflikte**
5. **Aufgabe 3: Projekt erstellen**
6. **SVN-Server**
7. **Workflow für Lehrer**
8. **Erfahrungen**
9. **Anwendungsbeispiel Entwurfsmuster MVC**

- a) Legen Sie ein neues BlueJ-Projekt mit einem beliebigen Namen an.
- b) Laden Sie es mit dem Menüpunkt **Werkzeuge/Teamarbeit /Projekt gemeinsam nutzen...** in das Repository hoch.
- c) Schauen Sie dann, ob andere Teilnehmen schon ein BlueJ-Projekt hochgeladen haben und checken Sie es aus.



- Repositories können nicht durch BlueJ erzeugt werden



- Repositories können nicht durch BlueJ erzeugt werden
- Ein Repository kann und sollte mehrere BlueJ-Projekte enthalten
- Der BlueJ-SVN-Client kann BlueJ-Projekte im Repository erzeugen und verändern, aber nicht löschen

# Überblick

1. **Grundlegendes zu Versionsverwaltung**
2. **Aufgabe 1, Teil 1: Auschecken**
3. **Aufgabe 1, Teil 2: Abgeben und Aktualisieren**
4. **Aufgabe 2: Konflikte**
5. **Aufgabe 3: Projekt erstellen**
6. **SVN-Server**
7. **Workflow für Lehrer**
8. **Erfahrungen**
9. **Anwendungsbeispiel Entwurfsmuster MVC**

Zum Beispiel: SourceForge, riouxsvn.com

Vorteile: kostenlos

Nachteile: öffentlich, User müssen einzeln angelegt werden



☒ **Wiki**

Documentation is key to your project and the wiki tool helps make it easy for anyone to contribute.



☐ **Files & Stats**

Use the largest free, managed, global mirror network to distribute your files, and follow the download trends that enable you to develop better software.



☐ **Git**

Git is a distributed version control system designed to handle everything from small to very large projects with speed and efficiency.



☒ **Tickets**

Bugs, enhancements, tasks, etc., will help you plan and manage your development.



☒ **Forums**

Collaborate with your community in your forum.



☐ **Blog**

Share exciting news and progress updates with your community.



☒ **SVN**

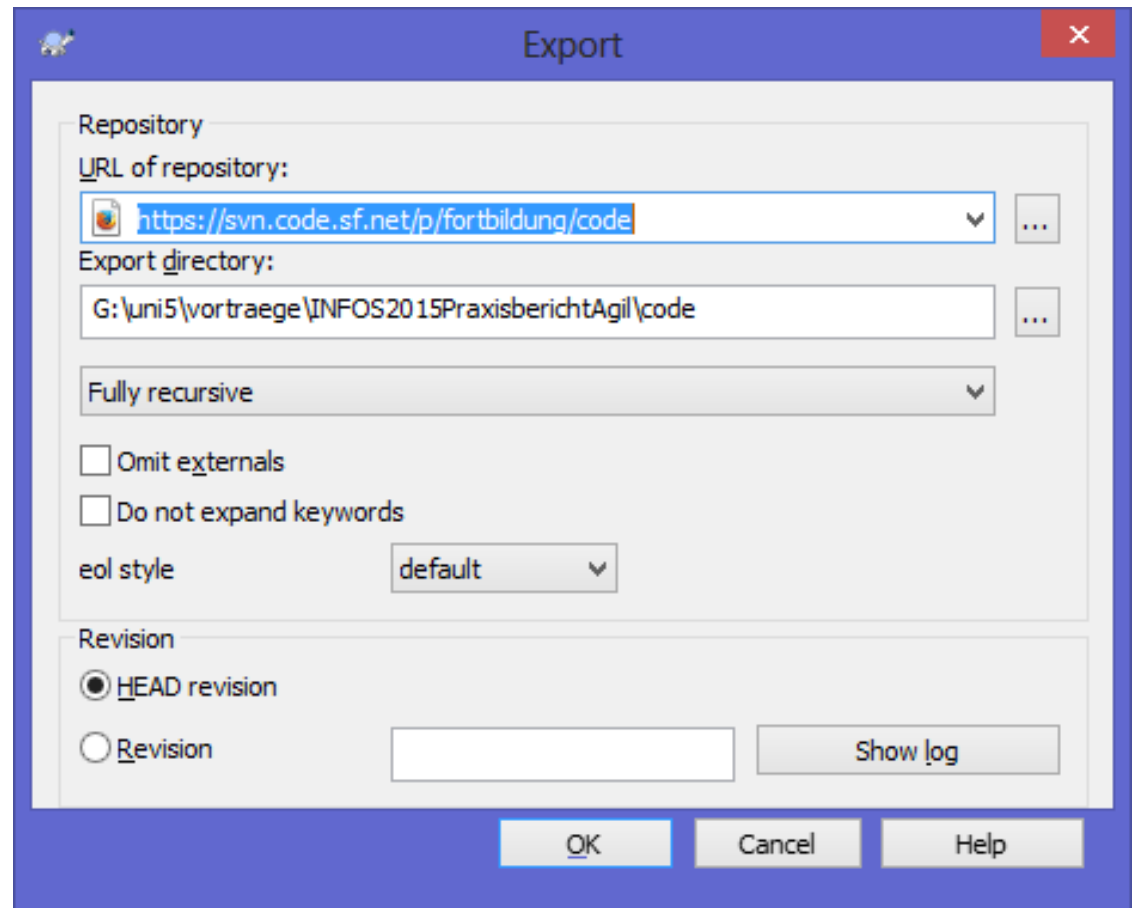
Enterprise-class centralized version control for the masses.



☐ **Mercurial**

Mercurial is a distributed source control management tool that efficiently handles projects of any size and offers an easy and intuitive interface.

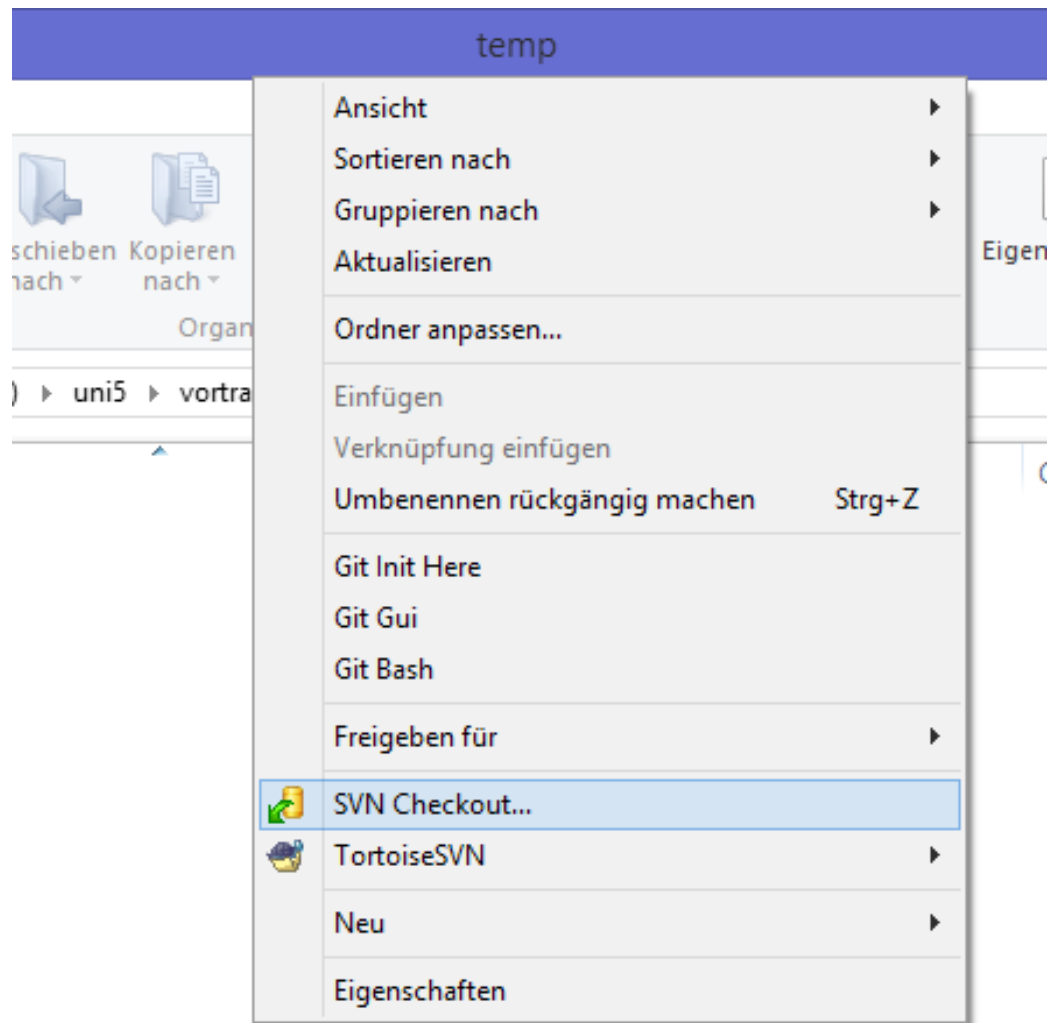
Tortoise SVN –  
Auswahl von zu  
synchronisierenden  
Verzeichnissen
















SVN Client:  
Tortoise SVN –

Checkout über  
Kontextmenü in der  
Dateiverwaltung

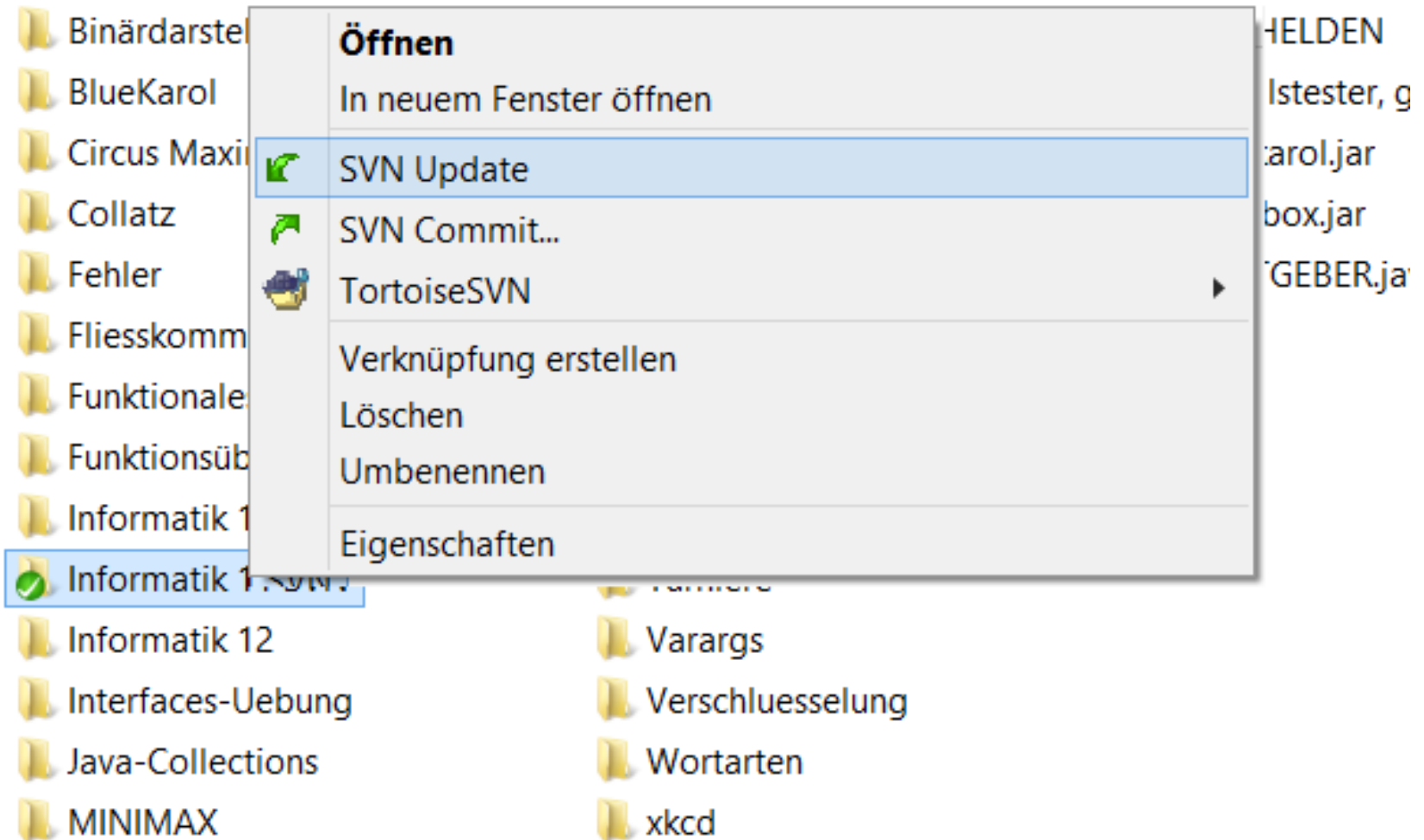


Tortoise SVN –  
Auswahl von zu  
synchronisierenden  
Verzeichnissen

(die alle Teil des  
gleichen Repository  
sind)

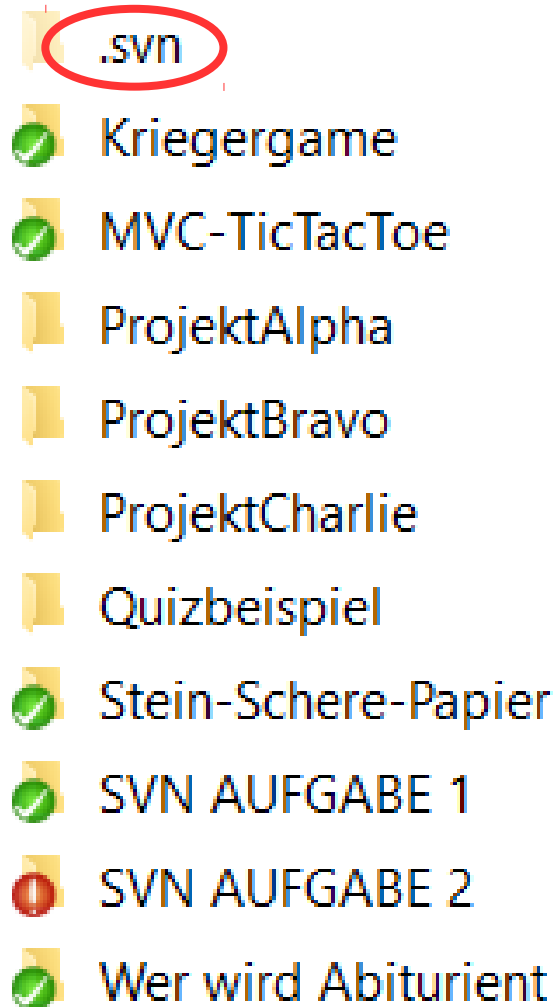
-  .svn
-  Kriegergame
-  MVC-TicTacToe
-  ProjektAlpha
-  ProjektBravo
-  ProjektCharlie
-  Quizbeispiel
-  Stein-Schere-Papier
-  SVN AUFGABE 1
-  SVN AUFGABE 2
-  Wer wird Abiturient

# Workflow für die Lehrkraft: Update und Commit

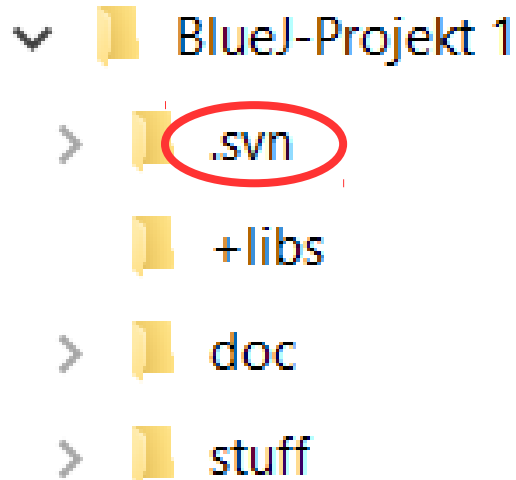


Tortoise SVN –  
Auswahl von zu  
synchronisierenden  
Verzeichnissen

(die alle Teil des  
gleichen Repository  
sind)



# Vorsicht: Bei Verwendung des BlueJ-SVN-Clients werden Daten *innerhalb* des Projekts gespeichert



Deshalb:

- Schüler und Schülerinnen arbeiten nur mit BlueJ
- Lehrer und Lehrerinnen arbeiten nur mit TortoiseSVN

# Überblick

1. **Grundlegendes zu Versionsverwaltung**
2. **Aufgabe 1, Teil 1: Auschecken**
3. **Aufgabe 1, Teil 2: Abgeben und Aktualisieren**
4. **Aufgabe 2: Konflikte**
5. **Aufgabe 3: Projekt erstellen**
6. **SVN-Server**
7. **Workflow für Lehrer**
8. **Erfahrungen**
9. **Anwendungsbeispiel Entwurfsmuster MVC**

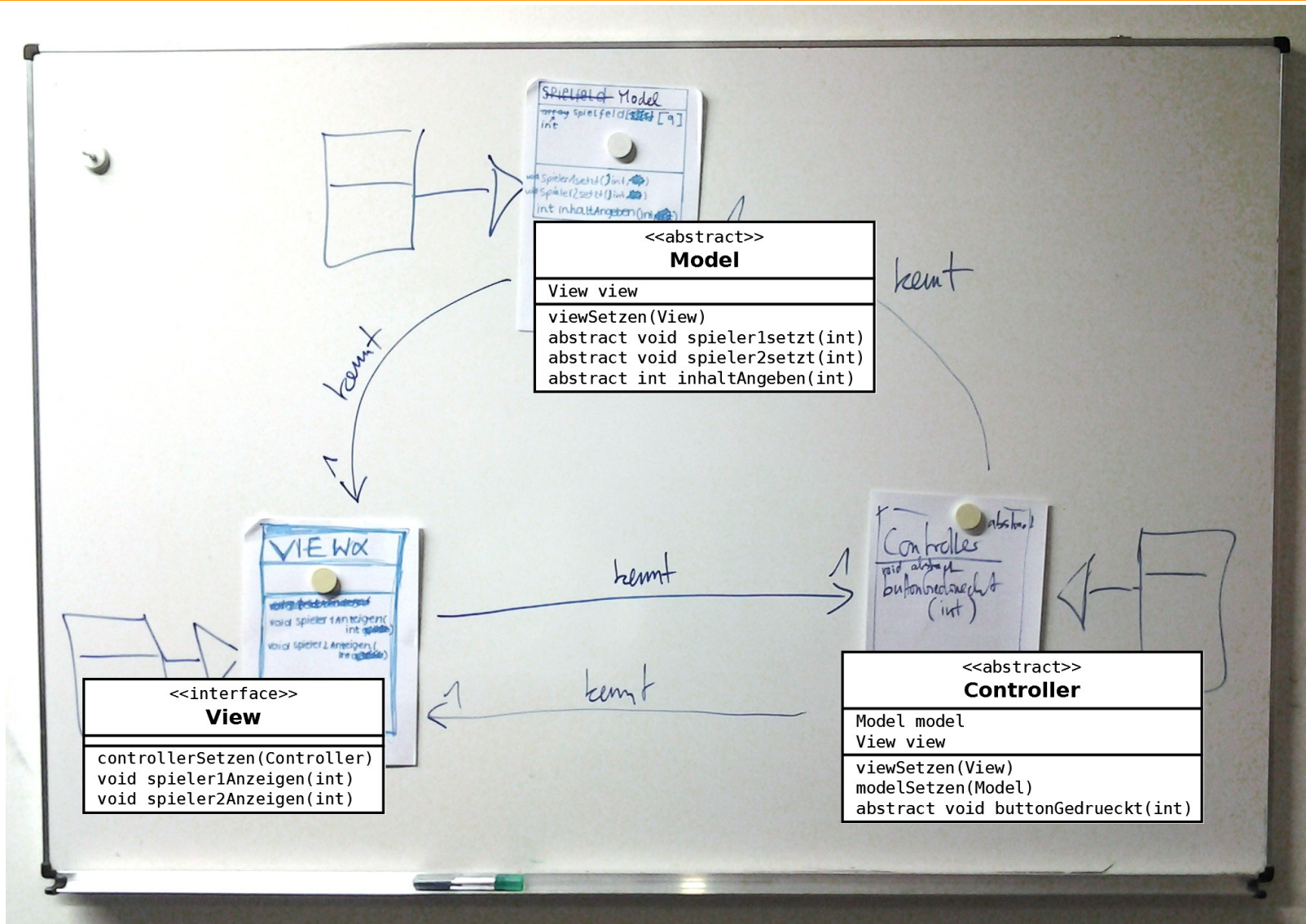
## Rückmeldungen aus der Praxis Typische Probleme

„If you get errors, save your work elsewhere, delete the project, and download a fresh copy.“



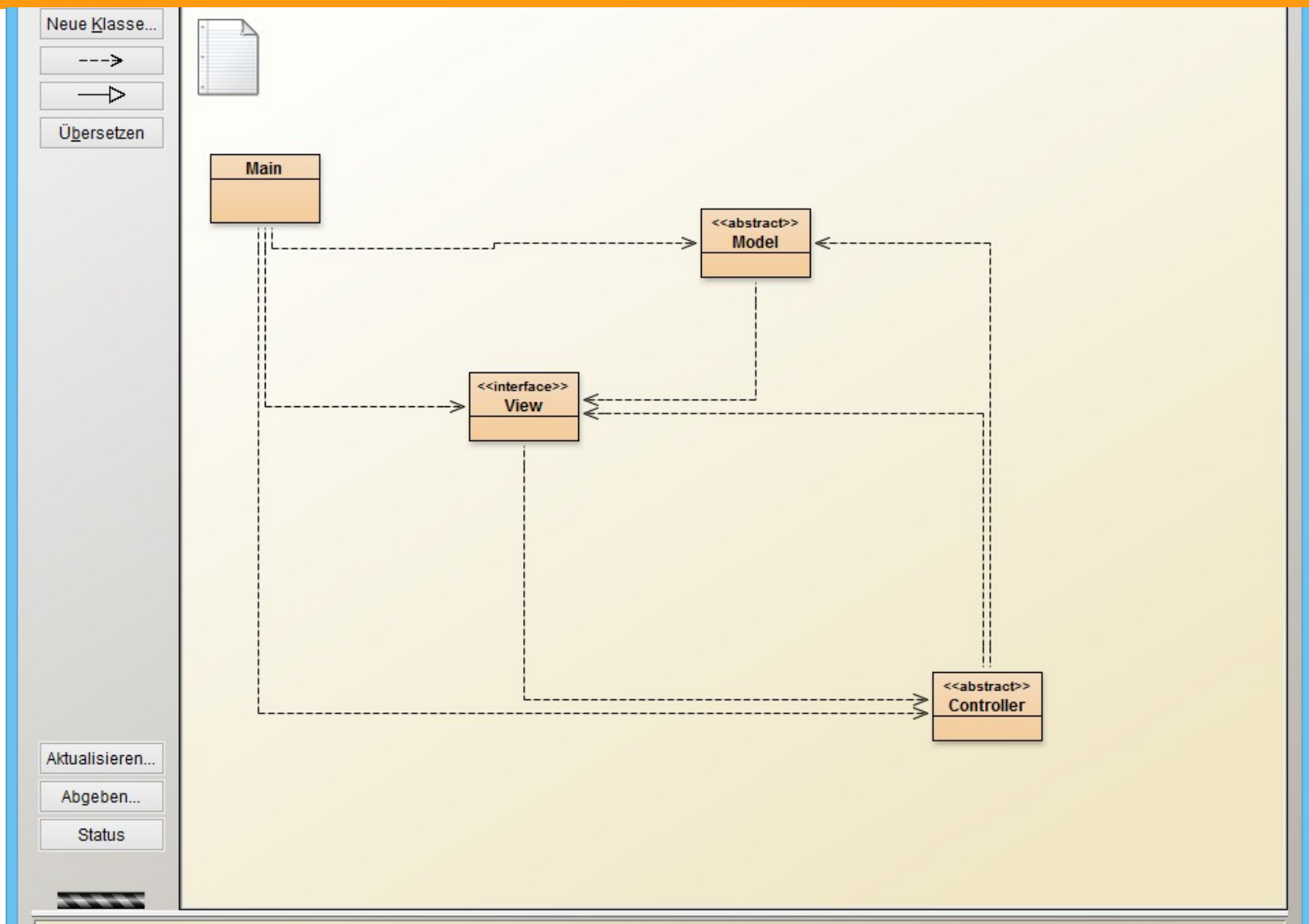
Randall Munroe/xkcd, Creative Commons Attribution-NonCommercial 2.5

# Anwendungsbeispiel Entwurfsmuster MVC

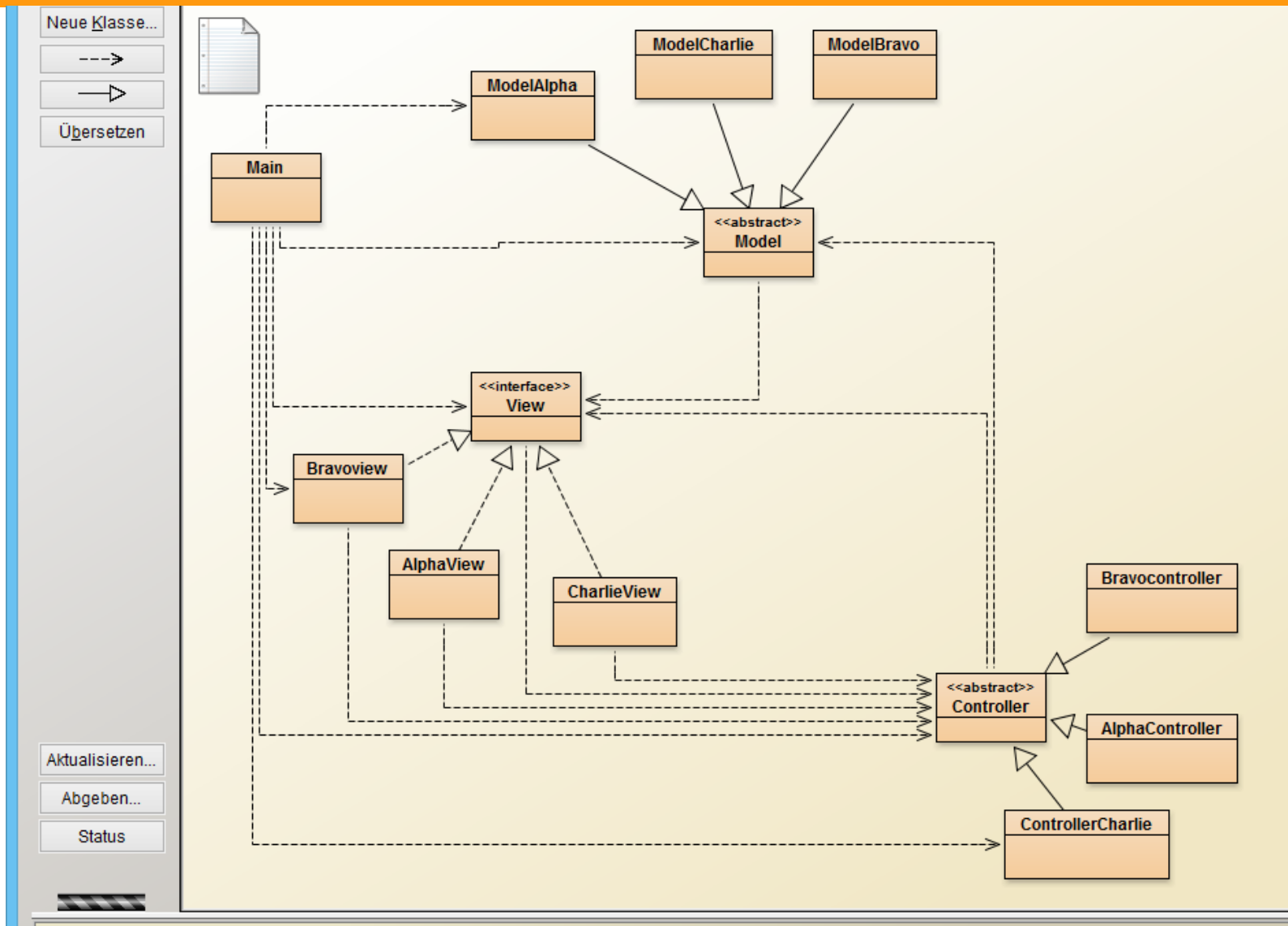


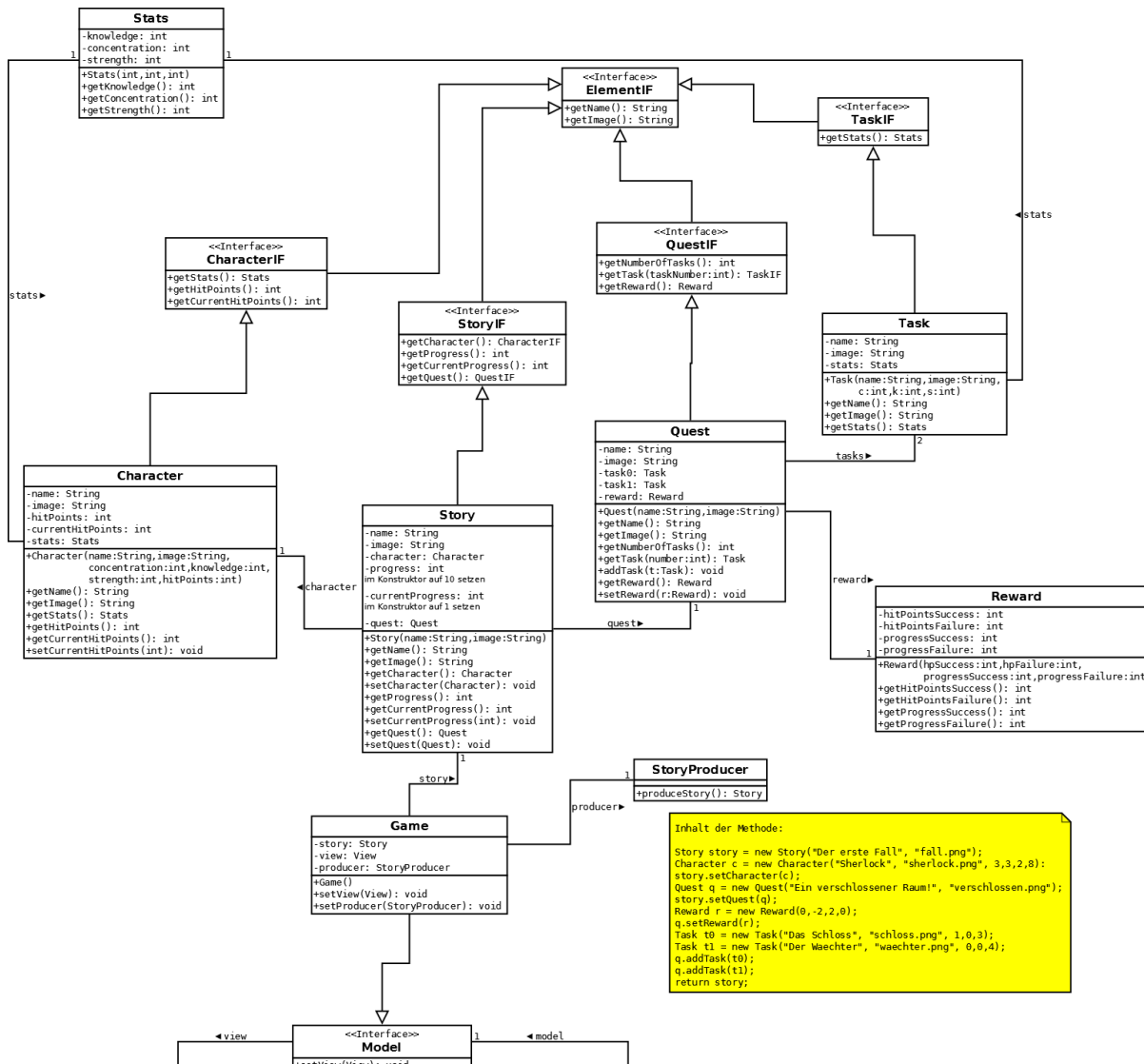


# Anwendungsbeispiel Entwurfsmuster MVC



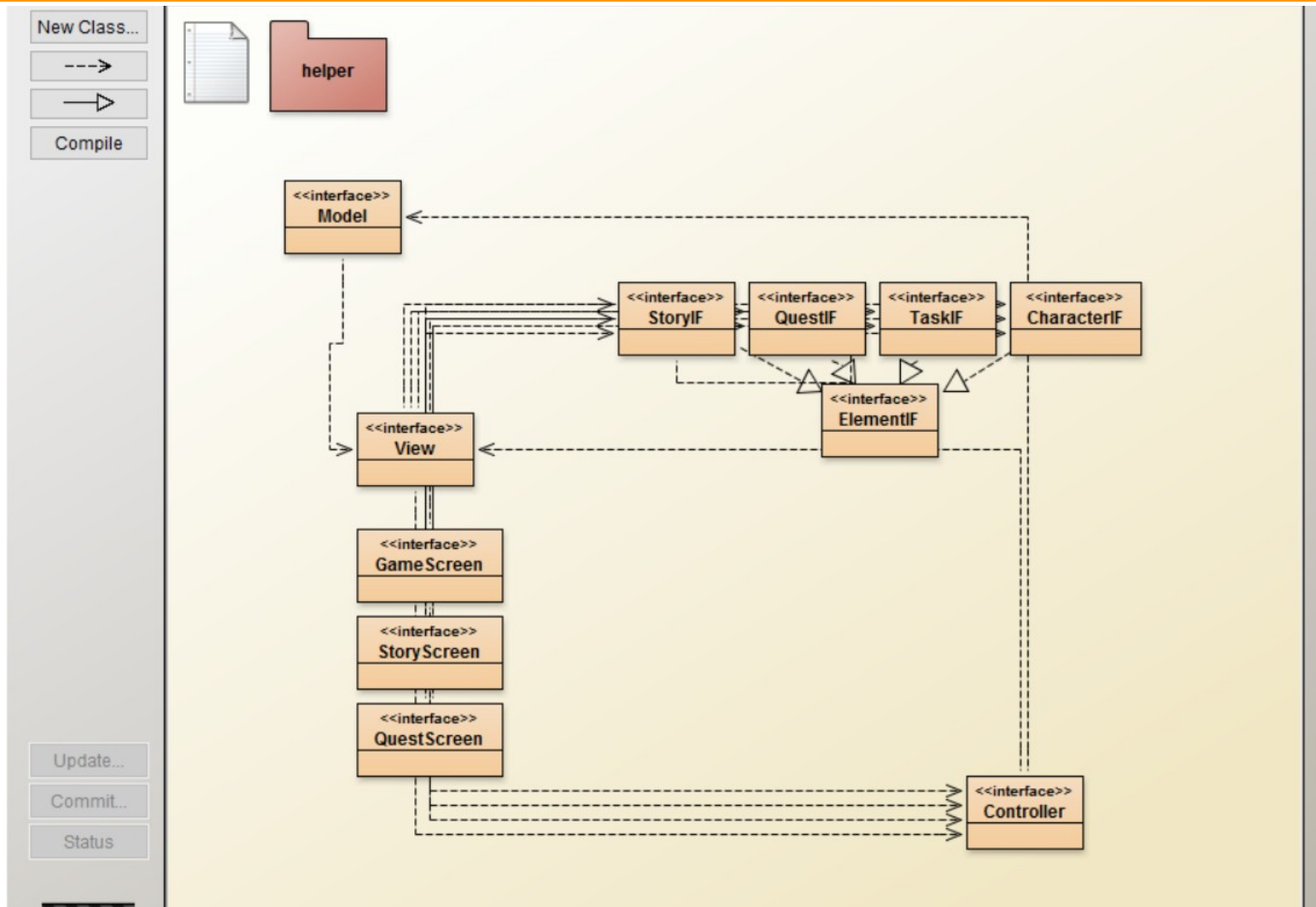
# Anwendungsbeispiel Entwurfsmuster MVC

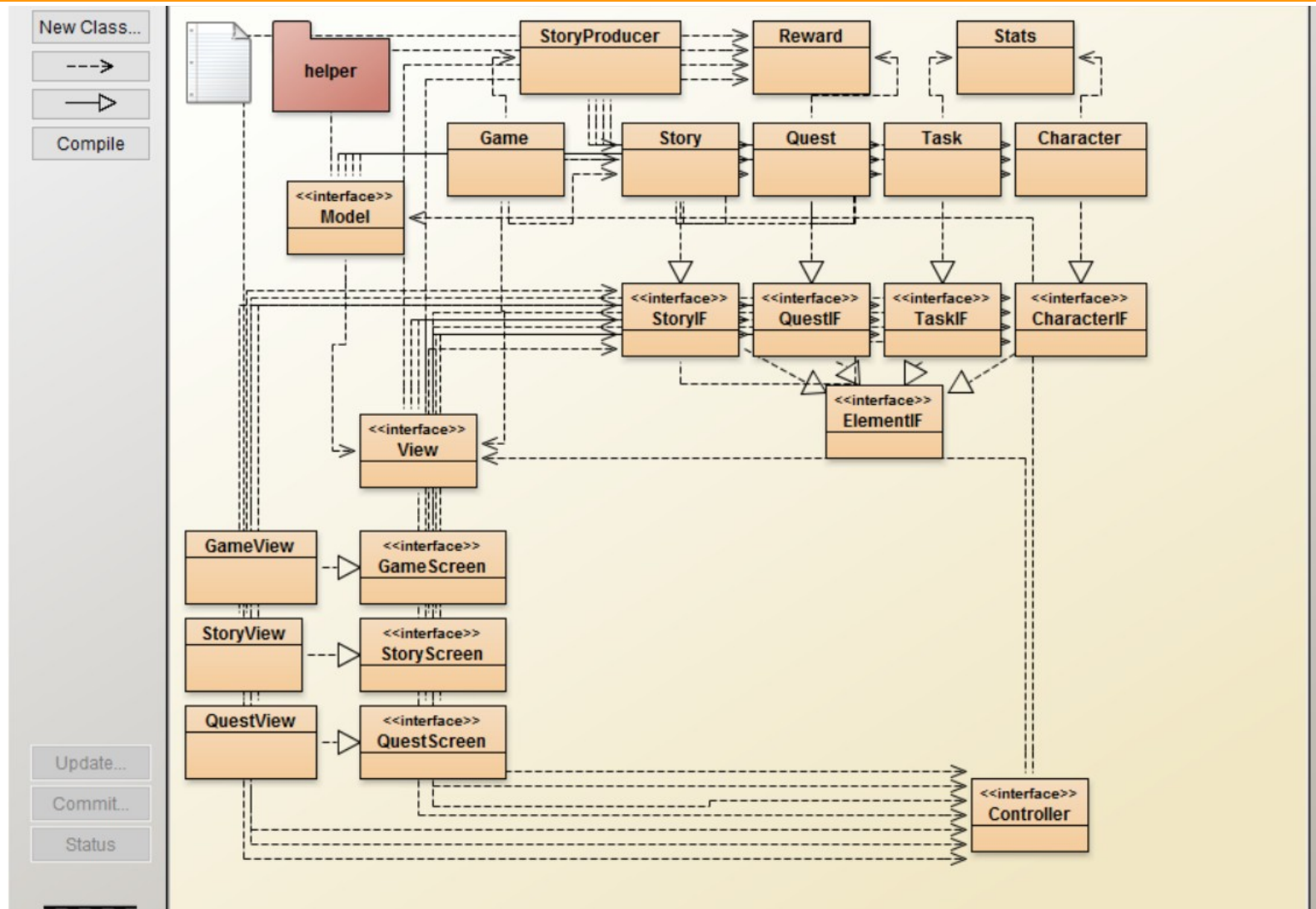




```

Inhalt der Methode:
Story story = new Story("Der erste Fall", "fall.png");
Character c = new Character("Sherlock", "sherlock.png", 3,3,2,8);
story.setCharacter(c);
Quest q = new Quest("Ein verschlossener Raum!", "verschlossen.png");
story.setQuest(q);
Reward r = new Reward(0,-2,2,0);
q.setReward(r);
Task t0 = new Task("Das Schloss", "schloss.png", 1,0,3);
Task t1 = new Task("Der Waechter", "waechter.png", 0,0,4);
q.addTask(t0);
q.addTask(t1);
return story;
    
```





# Fazit

# Versionskontrollsysteme bieten bei (Software-entwicklungs-)Projekten einen großen Mehrwert

- In Softwareprojekten gibt es oft organisatorische Probleme, die der inhaltlichen Arbeit ablenken!  
Versionskontroll-Systeme bieten mit  
**Verteiltem Zugriff**  
**Versionierung**  
**Datensicherheit**  
**Automatischem Zusammenführen**  
Lösungen zu den Problemen.
- Repositories vereinfachen dem Lehrer das Verteilen von Daten.
- Mit Versionskontrollsysteme können Schülerinnen und Schüler deutlich und schnell erfahren, welchen **Gewinn kollaboratives Arbeiten** hat.

**Material:** <http://ddi.ifi.lmu.de/tdi/2016/svn-bluej>