

Fakultät für Informatik

Versionsverwaltung mit Git

PR1 WS 21/22

1924363 – Lilly Schumann

1821830 – Lena Oeser

08.07.2021, Mannheim





Gliederung

1. Was ist Versionsverwaltung?
2. Warum sollte man es nutzen?
3. Einbindung in Eclipse
 - (1) Erstes Repository auf GitHub anlegen
 - (2) Lokales Repository erstellen
 - (3) Kommunikation der Repos
 - (4) Verbinden/Klonen
4. Merge Conflicts und Branches



Was ist Versionsverwaltung?

- Ein System, welches Datei-Änderungen speichert
- Es werden Versionen der Datei(en) über die Zeit gespeichert
- Ältere Versionen können wiederhergestellt werden





Was ist Versionsverwaltung?

- Dateien werden in einem **Repository** abgespeichert, welches die verschiedenen Versionen enthält
- Bei dem Versionsverwaltungssystem **Github** wird das Repository auf einem Server dort abgelegt
- Jeder der an dem Projekt mitarbeitet, kann es bei sich lokal **klonen**
- Nachdem lokale Änderungen gemacht wurden, werden diese hochgeladen, sodass jeder Beteiligte sie sehen kann



Warum sollte man es nutzen?

- **Macht Zusammenarbeit einfacher**
 - alle arbeiten an einem Ort zusammen
 - Jeder sieht die Änderungen der anderen & hat immer alles (kein rumschicken über Discord / per Mail notwendig)

Showing 2 changed files with 51 additions and 30 deletions.

```
src/game/Game.java
@@ -62,8 +62,9 @@ public static void main(String[] args) {
62 62     this.rollButton = new JButton("Würfeln");
63 63     this.rollButton.setFont(new Font("SansSerif", Font.BOLD, 28));
64 64     this.rollButton.setBackground(Color.PINK);
65 +     this.rollButton.setPreferredSize(new Dimension(140,50));
66 +     rollButton.setBorder(new LineBorder(Color.BLACK, 3, true));
65 67     this.rollButton.setFocusPainted(false);
66 -     this.setPreferredSize(new Dimension(90,60));
67 68     JPanel rollPanel = new JPanel();
68 69     rollPanel.setLayout(new FlowLayout());
69 70     rollPanel.add(rollButton);
```

hinzugefügte Code-Zeilen

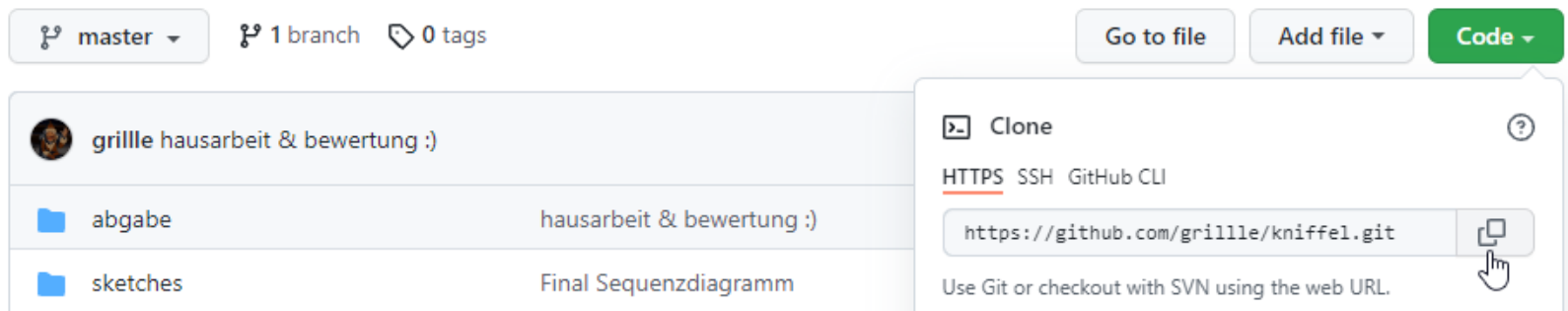
gelöschten Code-Zeilen

Beispiel aus GitHub



Warum sollte man es nutzen?

- **Fortschritt geht nicht verloren**
→ Wenn es lokal ein Problem gibt, kann es nochmal neu geklont werden



The screenshot shows a GitHub repository interface. At the top, there are buttons for 'Go to file', 'Add file', and 'Code'. Below these, the repository name 'grillle hausarbeit & bewertung :)' is displayed. A table lists the repository's contents:

File/Folder	Description
abgabe	hausarbeit & bewertung :)
sketches	Final Sequenzdiagramm

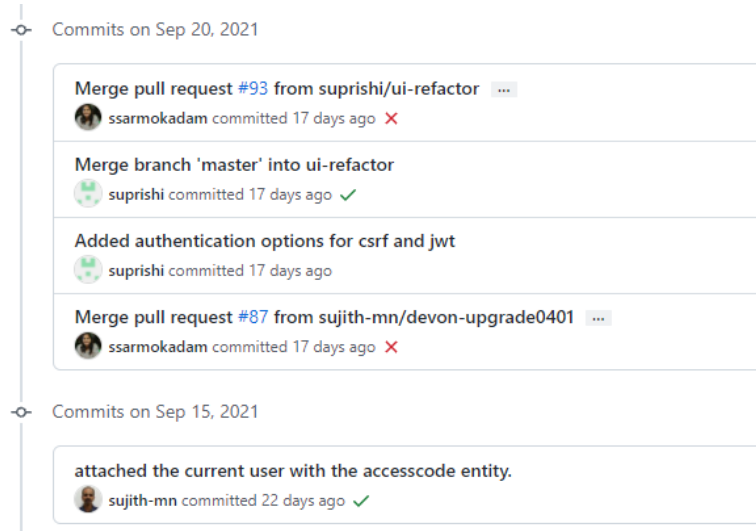
The 'Code' dropdown menu is open, showing options for cloning the repository. The 'Clone' button is selected, and the 'HTTPS' option is chosen. The URL `https://github.com/grillle/kniffel.git` is displayed in a text box, with a copy icon to its right. Below the text box, it says 'Use Git or checkout with SVN using the web URL.'

Beispiel aus GitHub



Warum sollte man es nutzen?

- **Im Notfall kann eine alte Version wieder hergestellt werden**
→ Alte Versionen werden mit abgespeichert und können leicht wieder hergestellt werden



Beispiel aus GitHub



Einbindung in Eclipse



1. Repository in GitHub anlegen

The screenshot shows the GitHub homepage with a dark theme. The 'Create repository' button is circled in red, and a red arrow points to it from the top left. The page includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and several informational cards. The 'Create your first project' card is highlighted with a red circle around the 'Create repository' button. Other cards include 'Learn Git and GitHub without any code!', 'Save the Date!', 'Introduce yourself', and 'Discover interesting projects and people to populate your personal news feed.'.

Search or jump to... Pull requests Issues Marketplace Explore

Create your first project
Ready to start building? Create a repository for a new idea or build over an existing repository to keep contributing to it.

Create repository Import repository

Recent activity
When you take actions across GitHub, we'll provide links to that activity here.

Learn Git and GitHub without any code!
Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Read the guide Start a project

Save the Date!
GitHub Universe is coming October 27 and 28. From product deep dives to interactive roundtables, you'll gather the tips, tools, and connections to help you do the best work of your life.

Learn more

All activity

Introduce yourself
The easiest way to introduce yourself on GitHub is by creating a README in a repository about you! You can start here:

```
1821830 / README.md
```

```
1 - 👋 HI, I'm @1821830
2 - 🐼 I'm interested in ...
3 - 🦉 I'm currently learning ...
4 - 🤝 I'm looking to collaborate on ...
5 - 📧 How to reach me ...
6
```

Dismiss this Continue

Discover interesting projects and people to populate your personal news feed.
Your news feed helps you keep up with recent activity on repositories you watch or star and people you follow.

Explore GitHub

1. Repository in GitHub anlegen

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * / Repository name * **PR1Tutorium** ✓

Great repository names are short and memorable. Need inspiration? How about [fluffy-enigma](#)?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

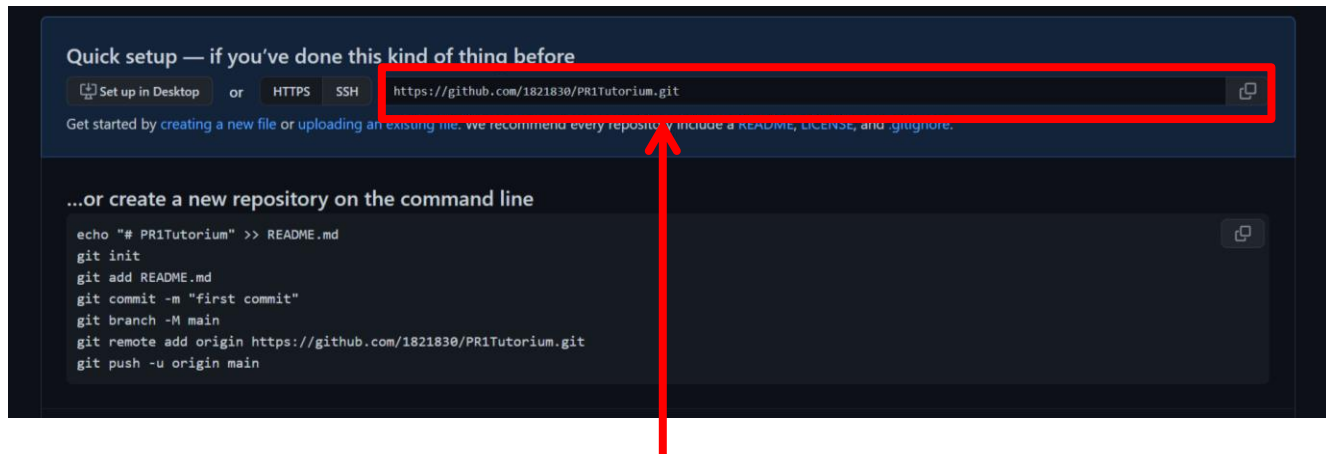
Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Namen eingeben

erstellen → fertig

1. Repository in GitHub anlegen

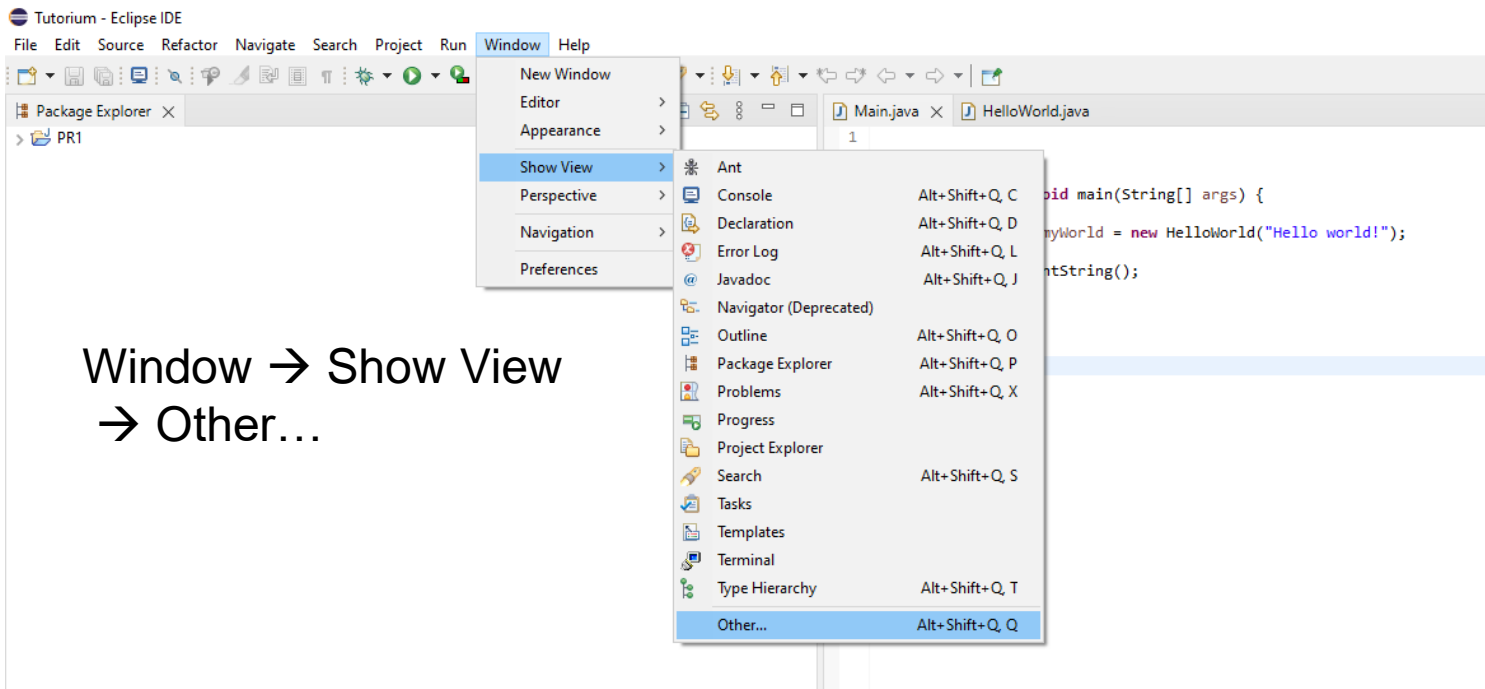


- Link zum eigenen Repository
- Wird später genutzt um
 - lokales Repository zu verknüpfen
 - bestehenden Stand zu klonen



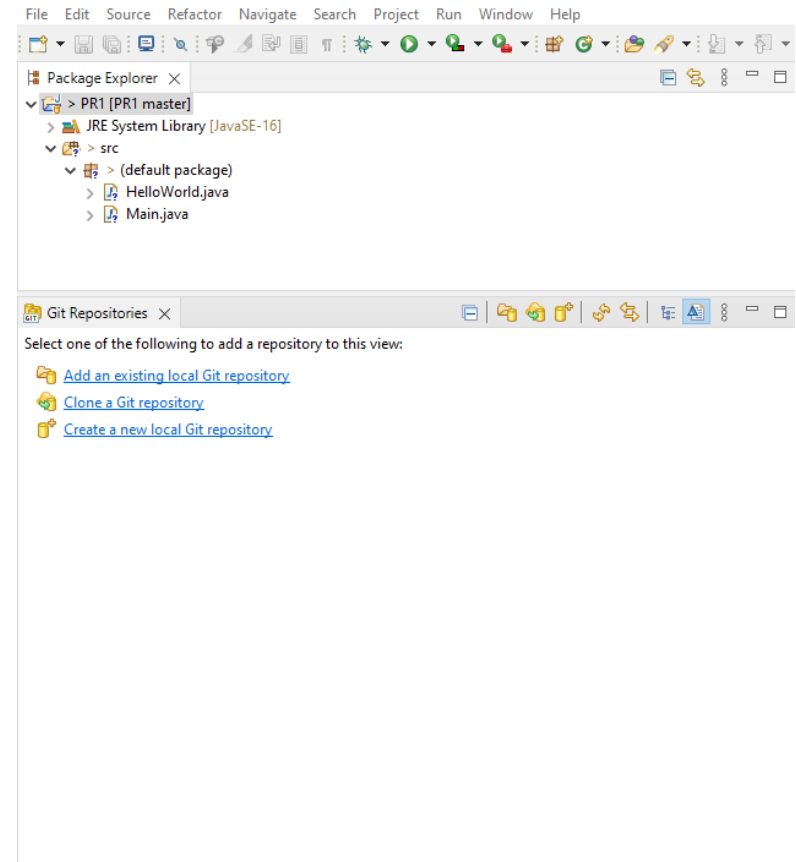
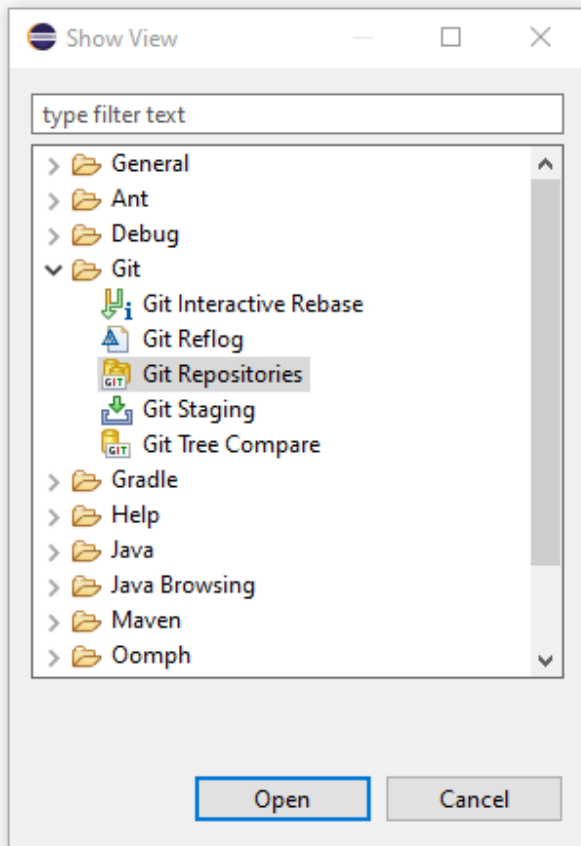
2. Lokales Repository erstellen

Erster Schritt: Git Ansicht in Eclipse öffnen

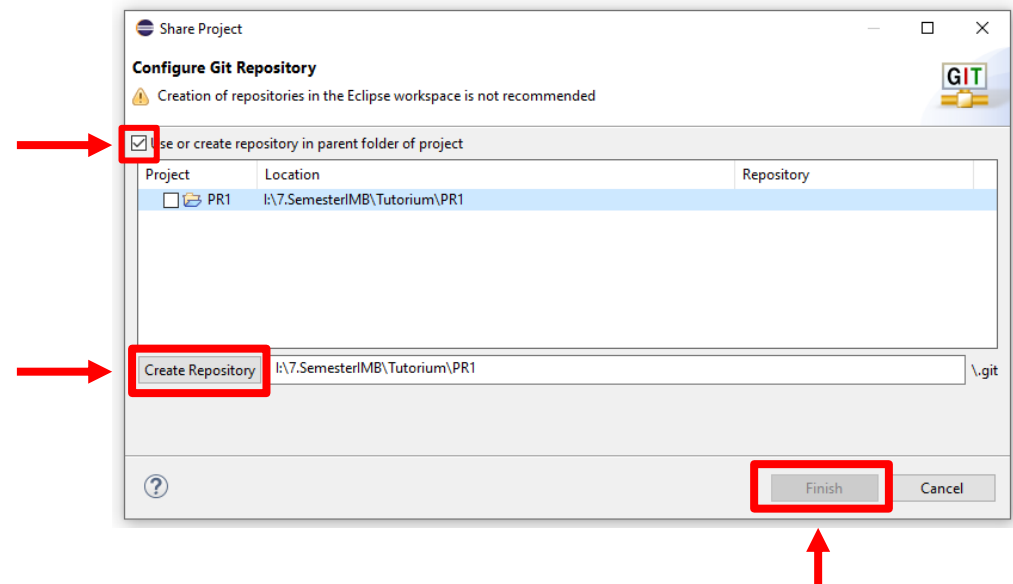
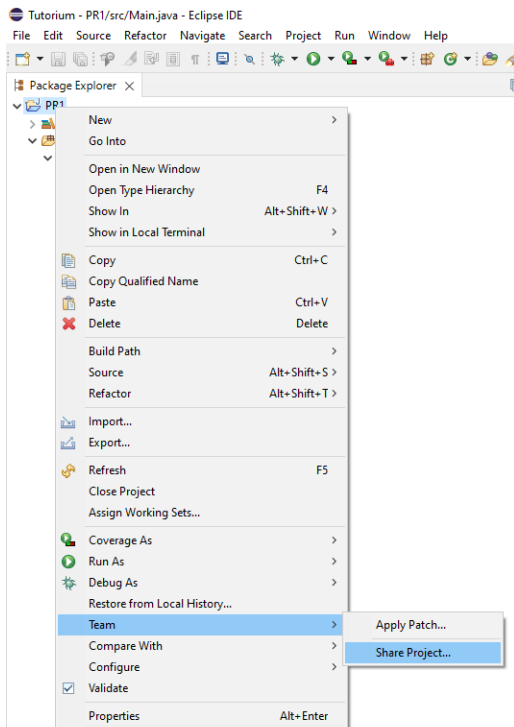


Window → Show View
→ Other...

2. Lokales Repository erstellen



2. Lokales Repository erstellen

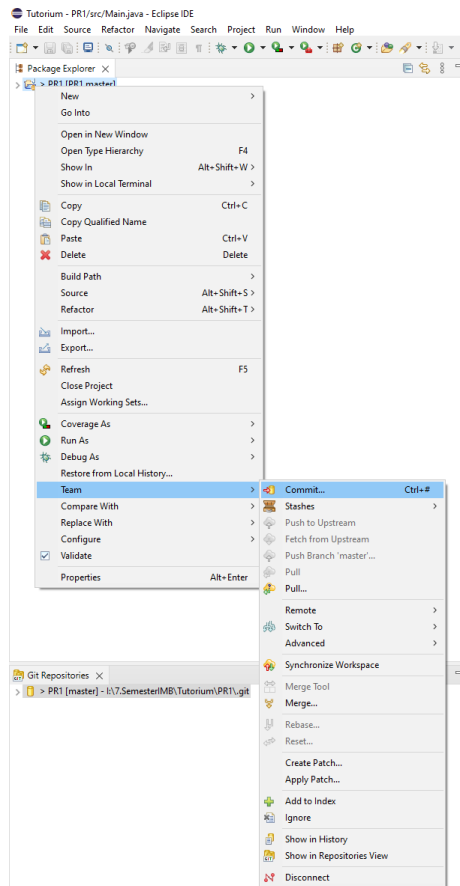


Rechtsklick auf Projekt →
Team → Share Project...

Checkbox auswählen
→ Repository erstellen
→ Fertigstellen

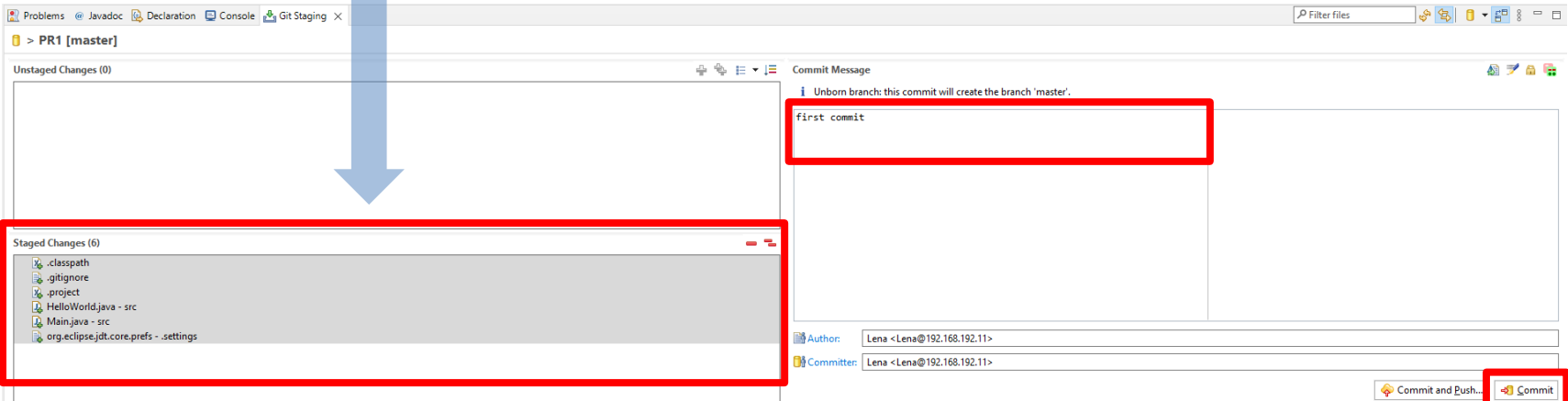
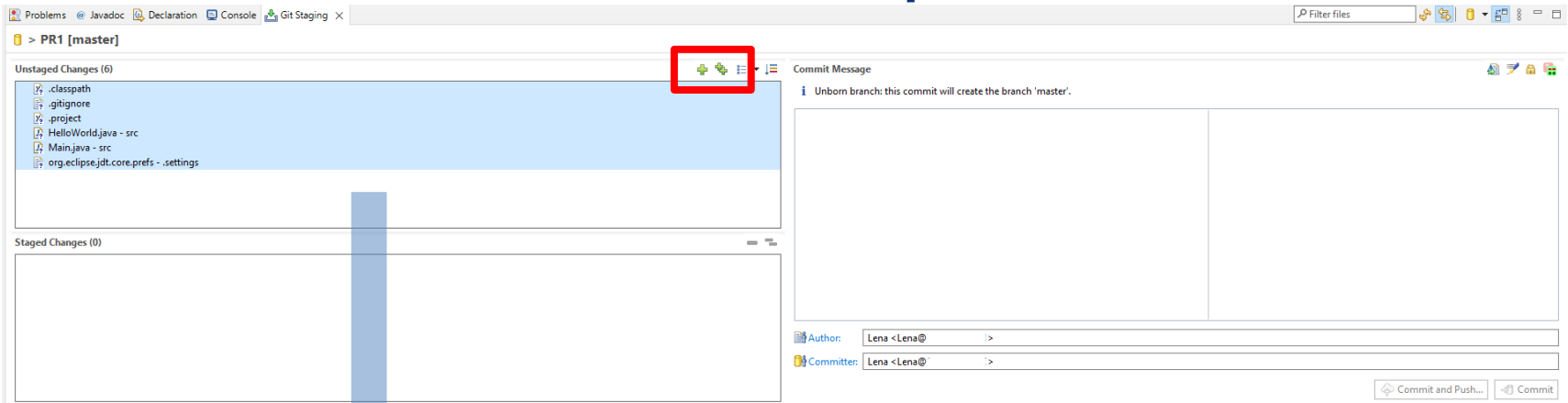


3. Kommunikation der Repositories – Commit



- Aktueller Stand des Projektes wird auf dem lokalen Repository festgehalten
- Öffnen mit Rechtsklick auf Projekt → Team → Commit...
- Ablauf:
 - Unstaged Dateien stagen
 - Dabei werden alle Dateien, welche im Commit enthalten sein sollen ausgewählt
 - => freie Zusammenstellung der Commits
 - Commit – Nachricht eingeben
 - Committen

3. Kommunikation der Repositories – Commit





3. Kommunikation der Repositories – Push

- Nach einem Commit kann nun gepusht werden
- Erst jetzt kommuniziert das lokale Repository mit dem online Repository
- Öffnen mit Rechtsklick auf Projekt → Team → Push Branch „master“
- Ablauf:
 - GitHub-Link vom Repository bei URI eingeben
 - Authentifikation über Github -> Daten eingeben und ggf. speichern

3. Kommunikation der Repositories – Push

Push Branch master

Destination Git Repository
Enter the location of the destination repository.

Remote name:

Location

URI: Local Folder...

Host:

Repository path:

Connection

Protocol:

Port:

Authentication

User:

Password:

Store in Secure Store

Push Branch master

Push to branch in remote
Select a remote and the name the branch should have in the remote.

Source:

Destination:

Remote:

Branch:

Configure upstream for push and pull

When pulling:

Force overwrite branch in remote if it exists and has diverged

Show [advanced push](#) dialog

Push Branch master

Push Confirmation
Confirm following expected push result.

master → master [new branch]

Message Details

Repository <https://github.com/1821830/PR1Tutorium.git>

Cancel push if result would be different than above because of changes on remote

Show dialog with result only when it is different from the confirmed result above



3. Kommunikation der Repos – Push → GitHub

The screenshot shows a GitHub repository page for '1821830 / PR1Tutorium'. The repository is public and has 1 star, 0 forks, and 0 watchers. The main content area shows a commit by 'Lena' with the message 'Lena and Lena first commit' and a commit hash of 'bd808f'. The commit includes several files: '.settings', 'src', '.classpath', '.gitignore', and '.project', all of which were first committed 1 hour ago. There is a green 'Code' button and a 'Go to file' button. Below the file list, there is a prompt to 'Add a README' to help people understand the project. The right sidebar contains sections for 'About' (no description provided), 'Releases' (no releases published), 'Packages' (no packages published), and 'Languages' (Java 100.0%). The footer of the page includes copyright information for GitHub, Inc. and various links like Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About.



3. Kommunikation der Repositories – Pull

- mehrere Entwickler arbeiten an einem Projekt
- um Änderungen der anderen Entwickler zu erhalten
→ pull
- Öffnen mit Rechtsklick auf Projekt → Team → Pull
- Änderungen werden durch Mergen in das lokale Repository eingebunden
- In der Regel vollautomatisch (außer bei Merge-Conflicts)



4. Verbinden / Klonen

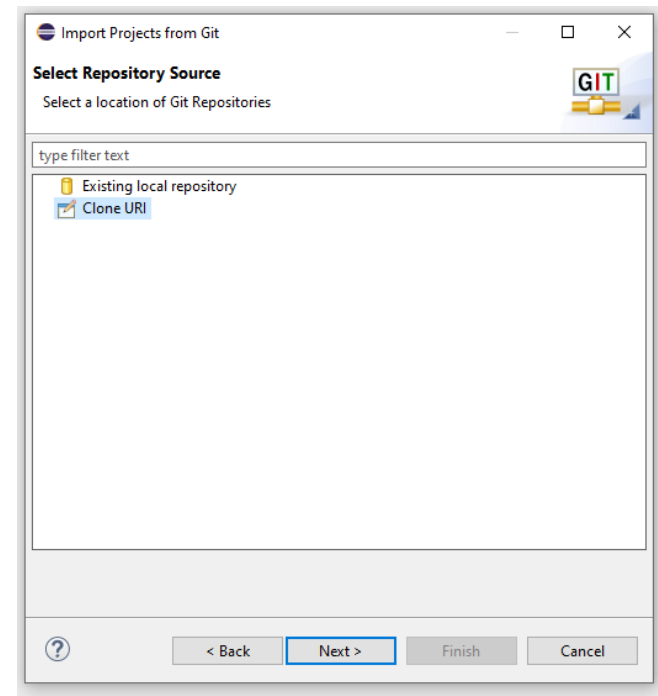
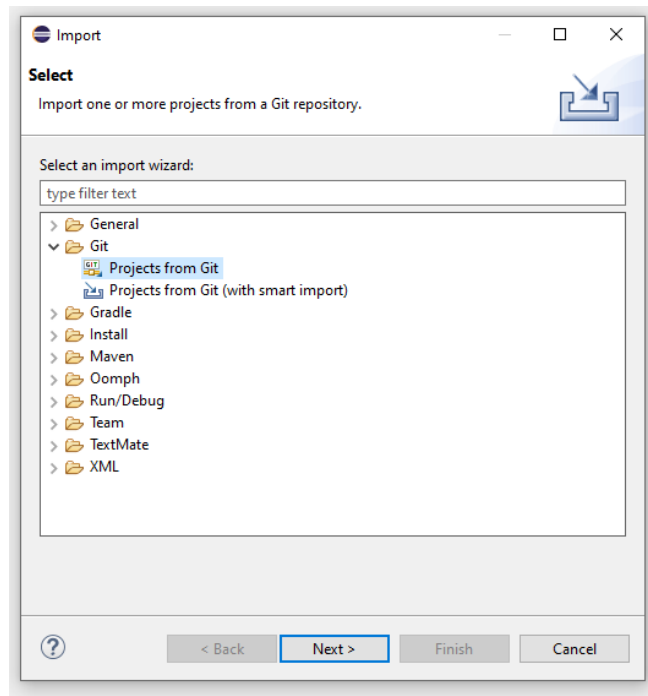
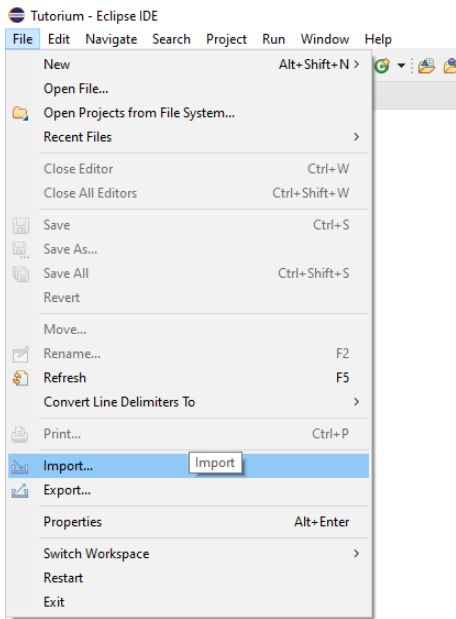
- Das lokale und das online Repository verbinden sich über Eclipse beim ersten Pull automatisch
- Ein weiterer Entwickler will Zugriff auf eigenes Projekt
→ Projekt muss geklont werden
- Aktueller Stand des online Repository wird vollständig geklont
→ Es kann direkt weiter gearbeitet werden

4. Klonen – Einladung in das Repository

The screenshot shows the GitHub interface for a repository named 'PRITutorium'. The 'Settings' tab is selected and highlighted with a red box. In the left sidebar, the 'Options' menu is open, and 'Manage access' is highlighted with a red box. The main content area shows 'Who has access' with 'PUBLIC REPOSITORY' and 'DIRECT ACCESS' sections. Below this, the 'Manage access' section is visible, with the 'Add people' button highlighted in red. A list of collaborators is shown, including 'lilly' with a 'Pending Invite' status. On the right side of the screenshot, there are four white arrows pointing to the 'Settings', 'Manage access', 'Add people', and 'Einladen mit GitHub Nutzernamen' elements.

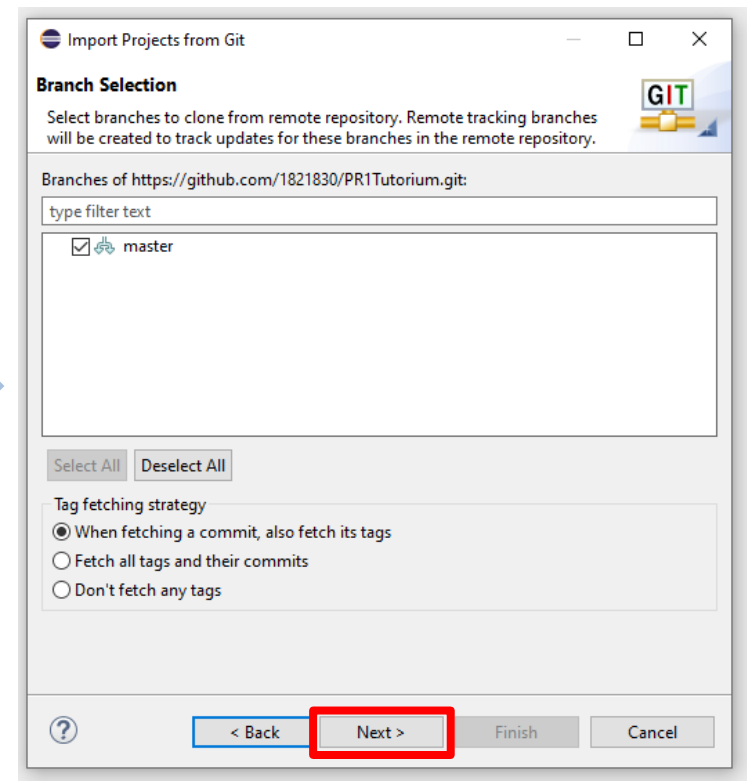
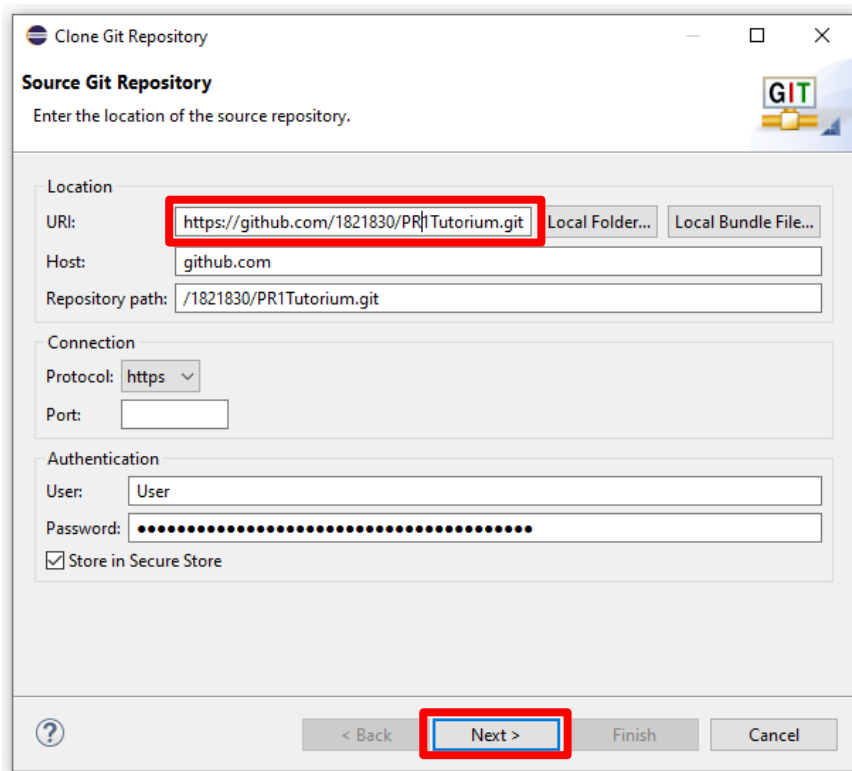
→ Settings
→ Manage access
→ Add people
→ Einladen mit GitHub Nutzernamen

4. Klonen in Eclipse

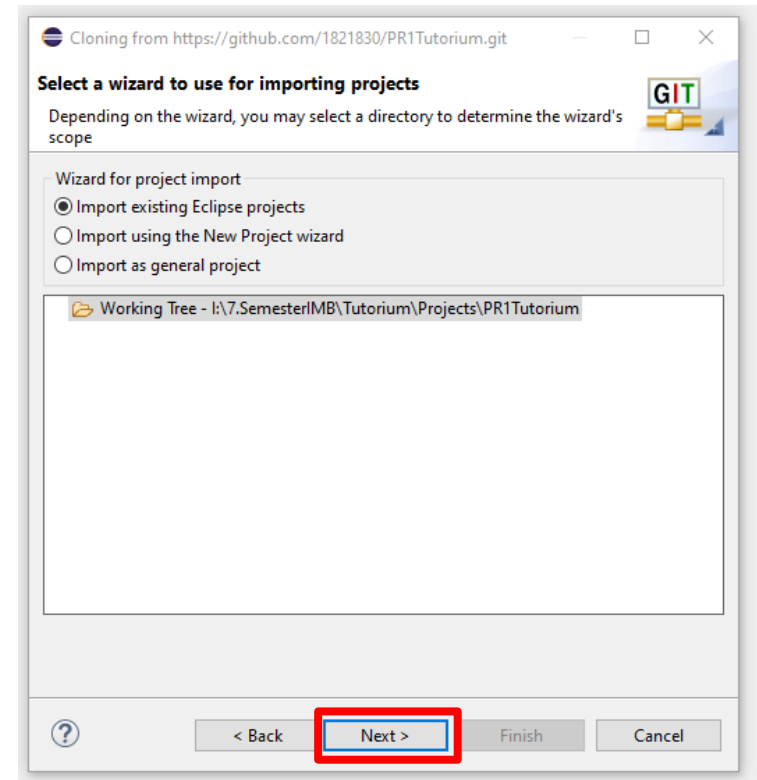
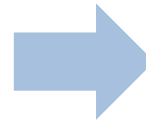
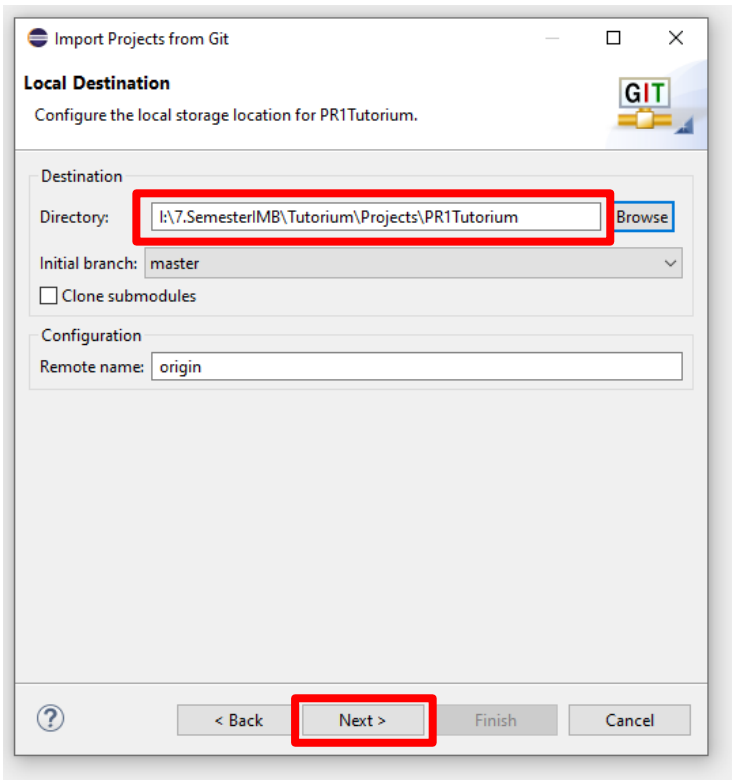


- File → Import...
- Projects from Git
- Clone URI

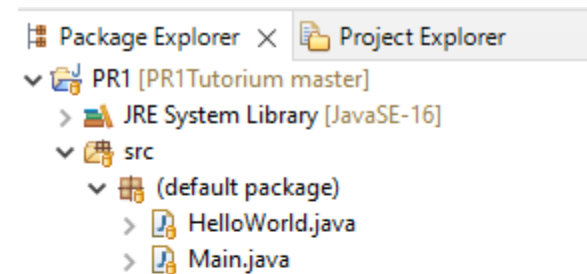
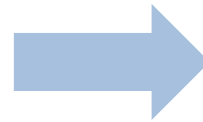
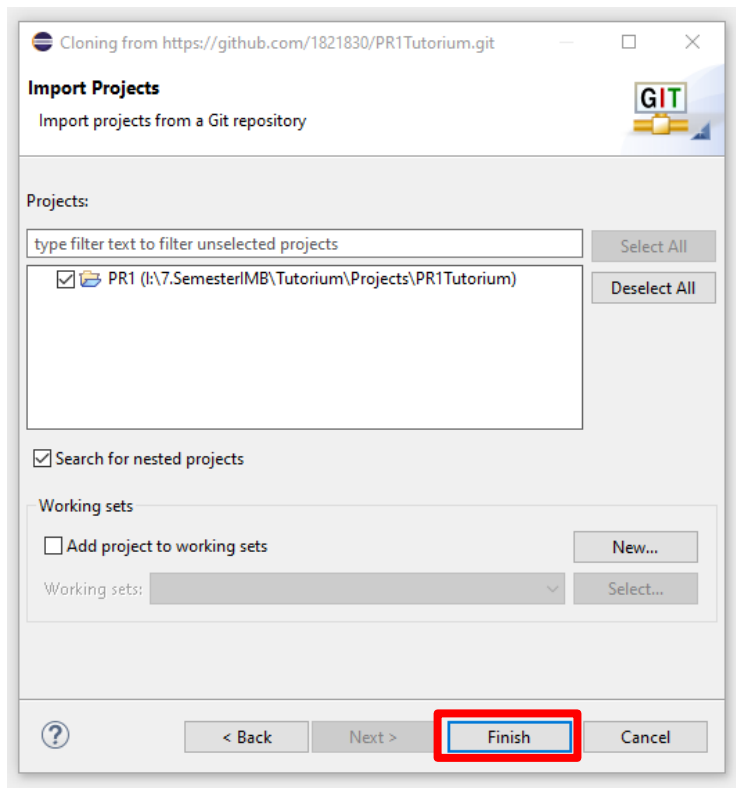
4. Klonen in Eclipse



4. Klonen in Eclipse



4. Klonen in Eclipse



→ Direkte Anbindung an das GitHub Repository

→ Voller Zugriff auf Repository



Merge Conflicts und Branches



Merge Conflicts

Falls mehrere Entwickler die exakt selbe Zeile unterschiedlich editieren, kann Git diese Änderungen nicht automatisch mergen
→ Merge Conflict

Dieser Konflikt muss manuell gelöst werden



Branches

- Weiterer Entwicklungspfad zuzüglich zum „master“ Branch
 - Separate Entwicklung → Mergen einmalig am Ende
- Vermeidung von ständigen manuellen Merges
→ Möglichkeit neue Features unabhängig zu entwickeln/testen

