hochschule mannheim

Fakultät für Informatik

Versionsverwaltung mit Git

PR1 WS 21/22

1924363 – Lilly Schumann

1821830 – Lena Oeser

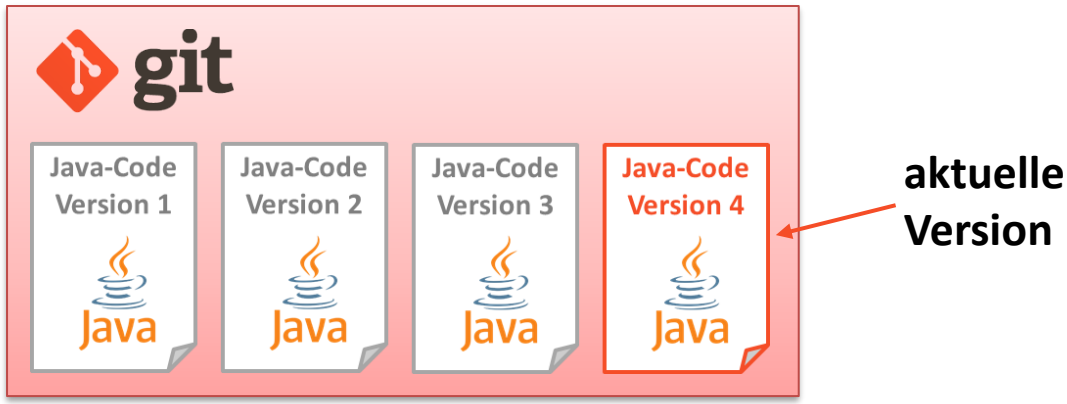
08.07.2021, Mannheim

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

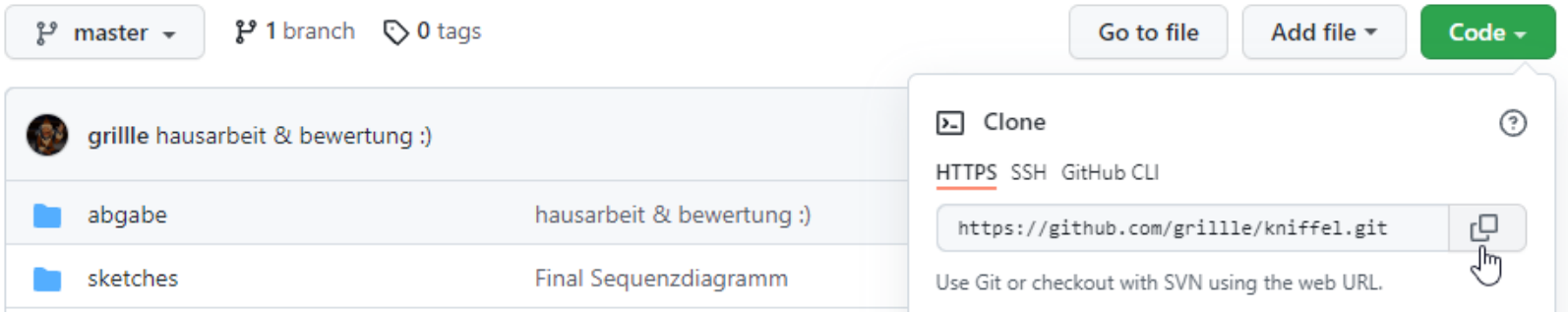
Warum sollte man es nutzen?

- **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)

Beispiel aus GitHub

Warum sollte man es nutzen?

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



Beispiel aus GitHub

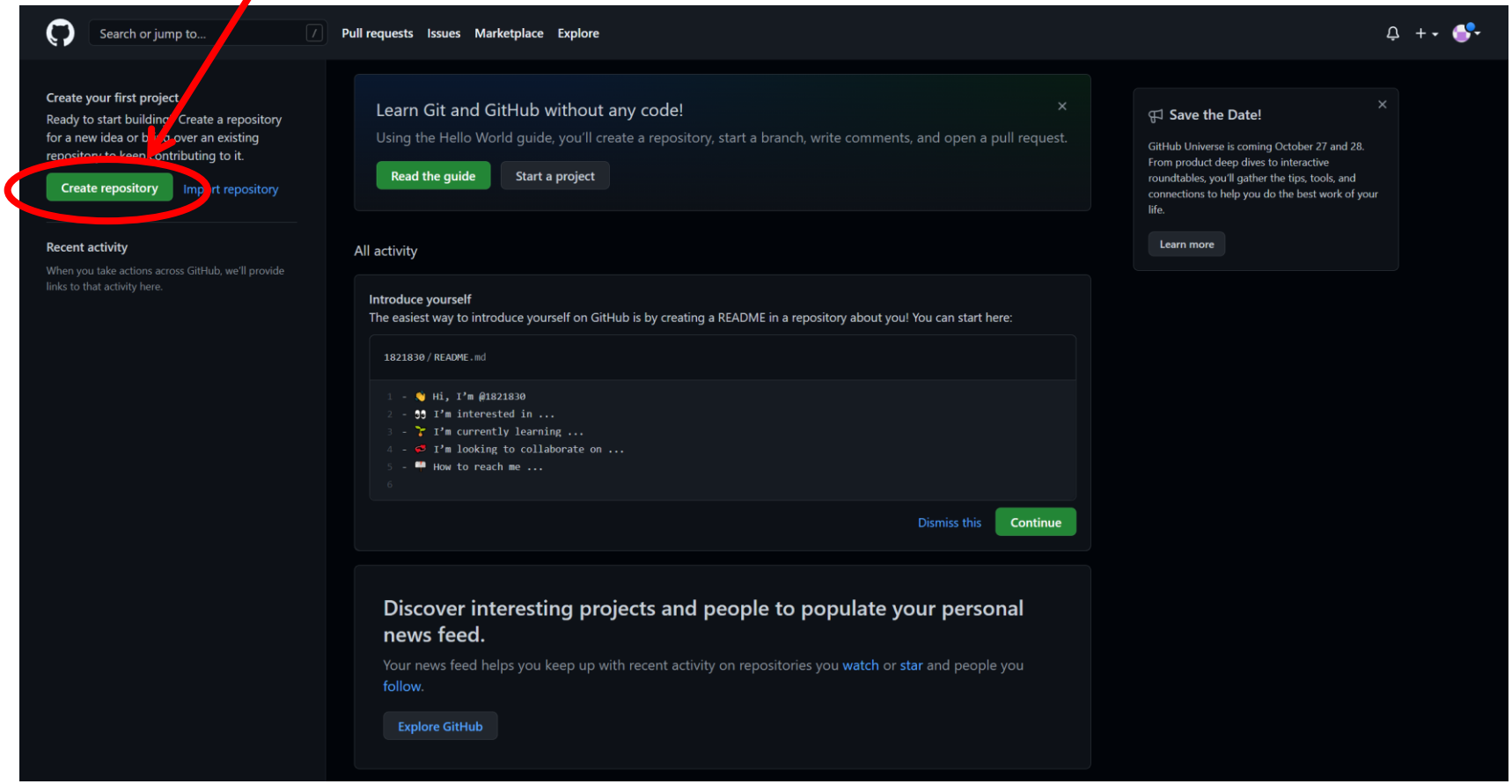
Warum sollte man es nutzen?

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

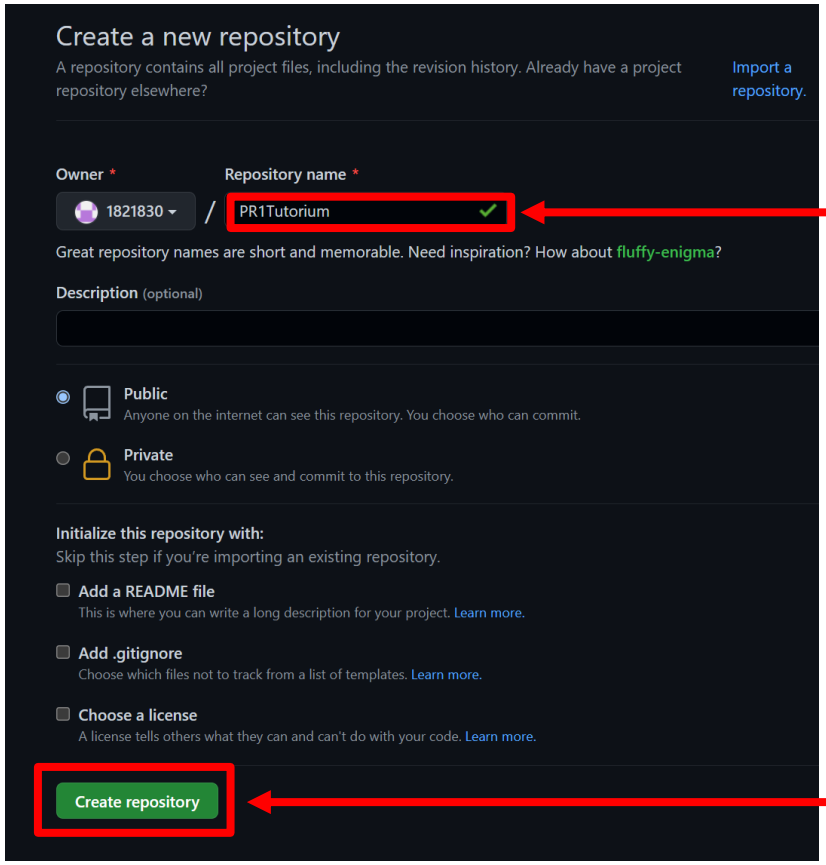
Beispiel aus GitHub

Einbindung in Eclipse

1. Repository in GitHub anlegen



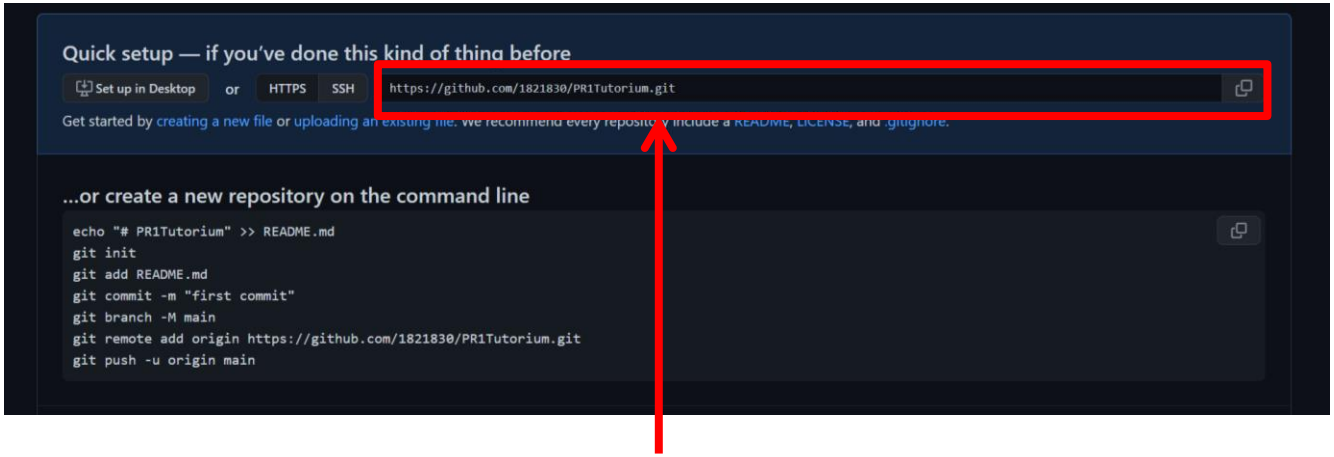
1. Repository in GitHub anlegen



Namen eingeben

erstellen → fertig

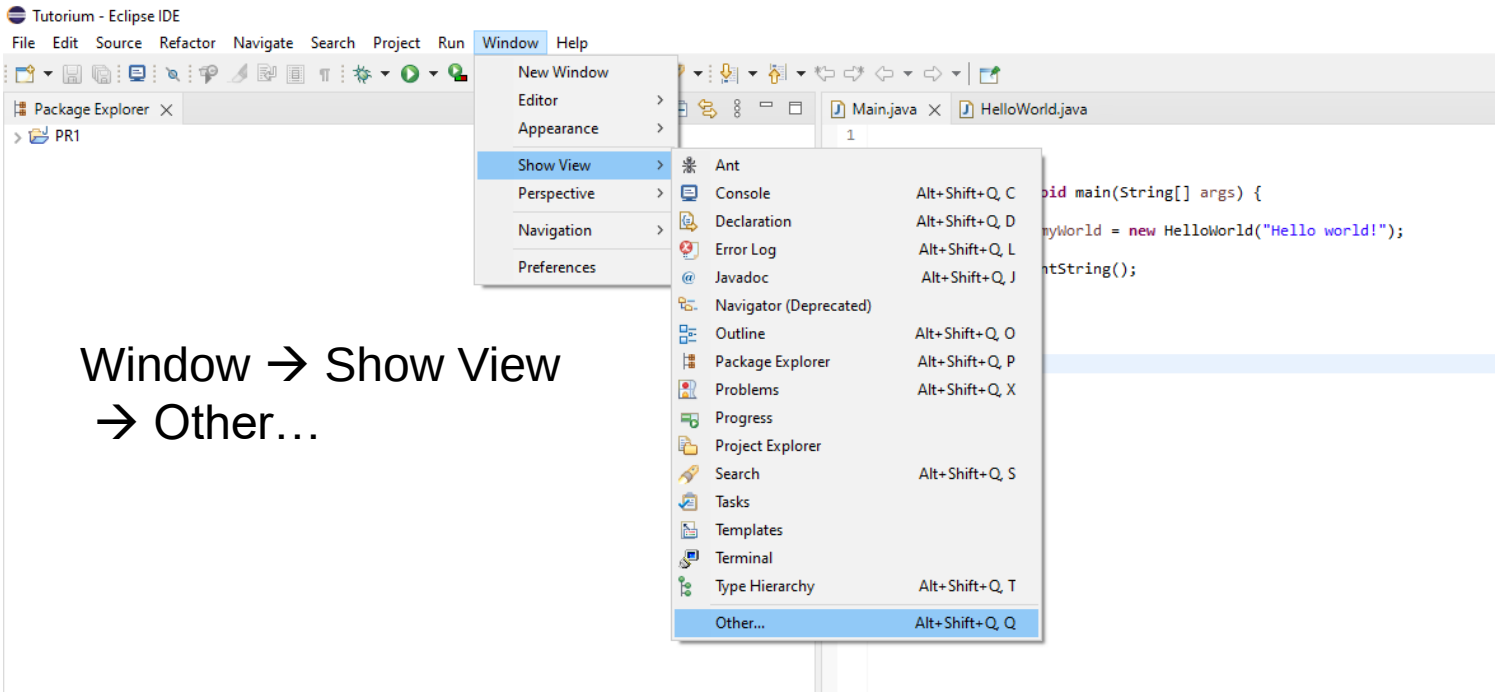
1. Repository in GitHub anlegen



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

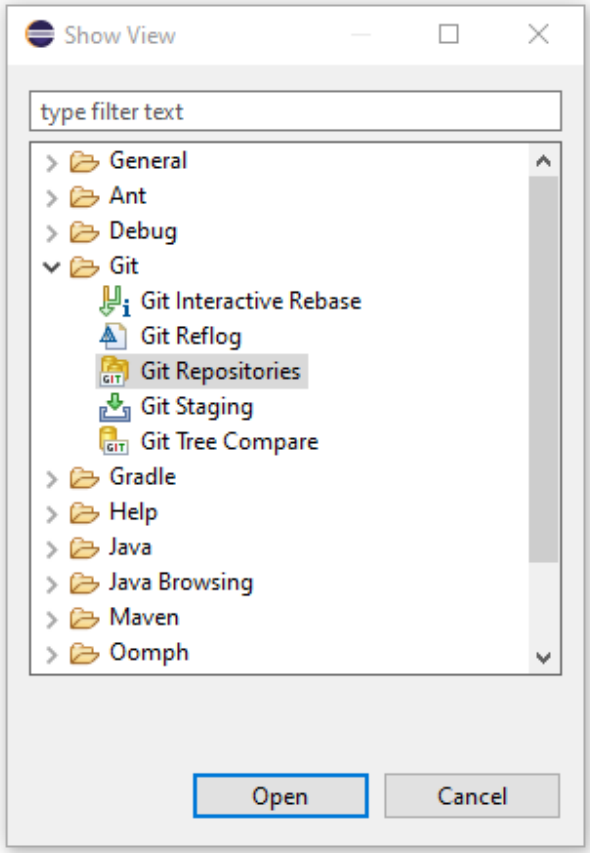
2. Lokales Repository erstellen

Erster Schritt: Git Ansicht in Eclipse öffnen

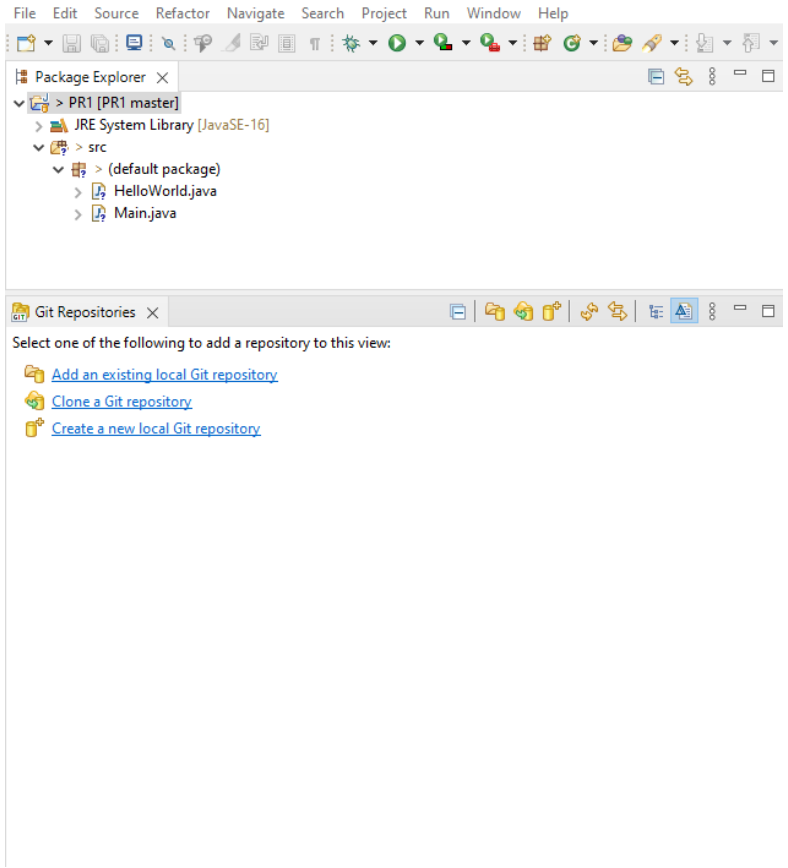


Window → Show View
→ Other…

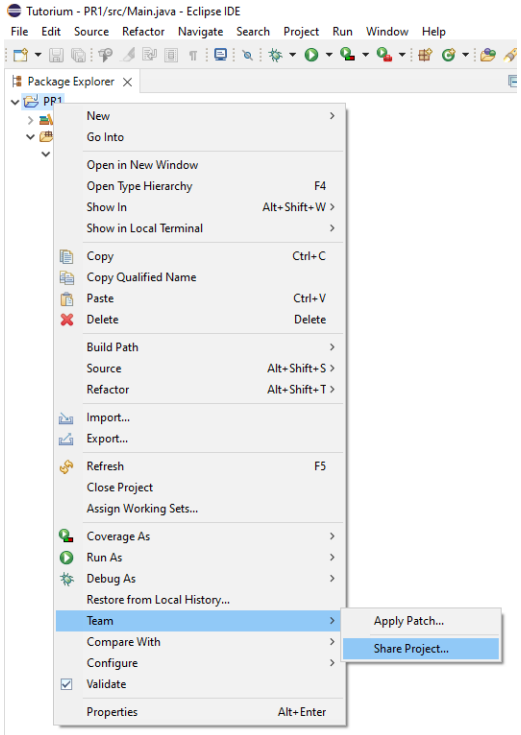
2. Lokales Repository erstellen



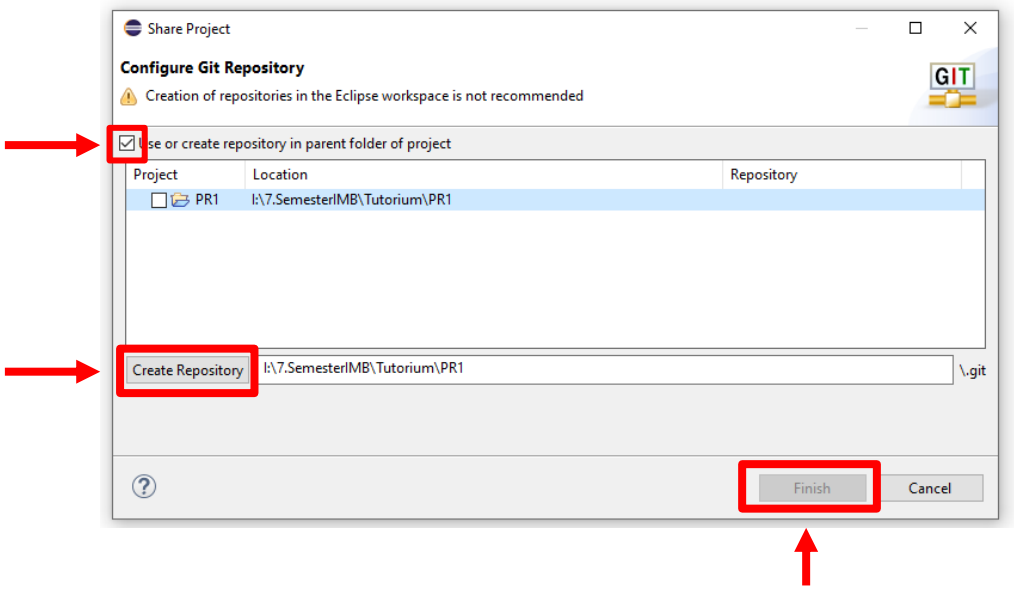
Git Ansicht



2. Lokales Repository erstellen

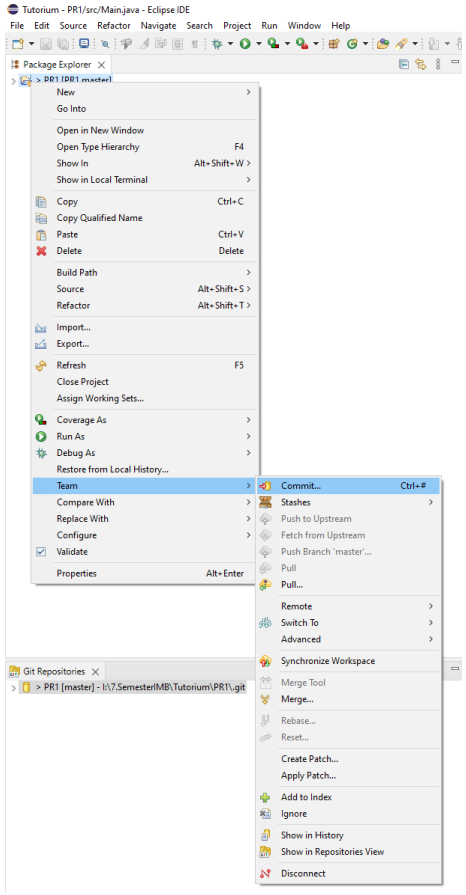


Rechtsklick auf Projekt → Team → Share Project…



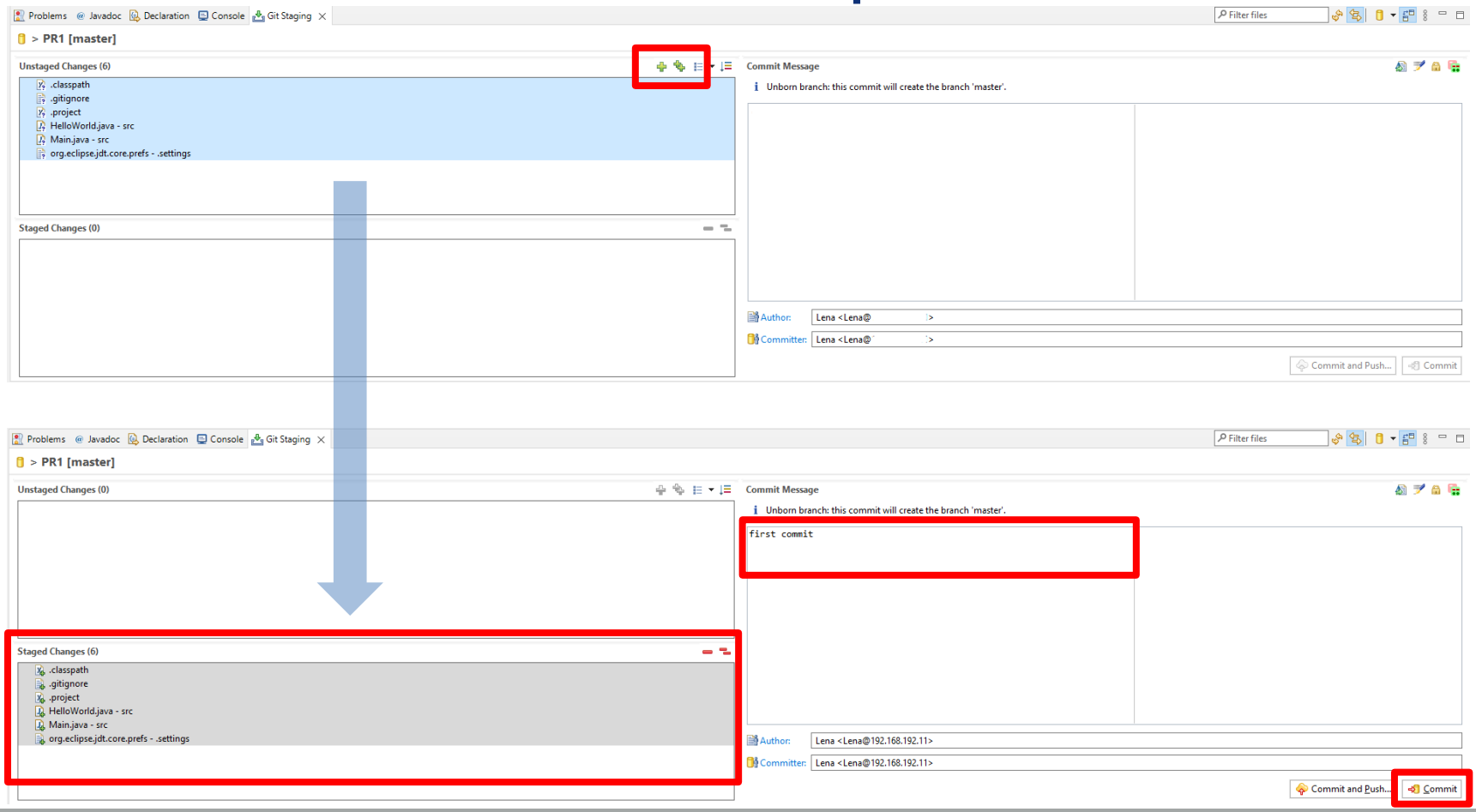
Checkbox auswählen
→ Repository erstellen
→ Fertigstellen

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

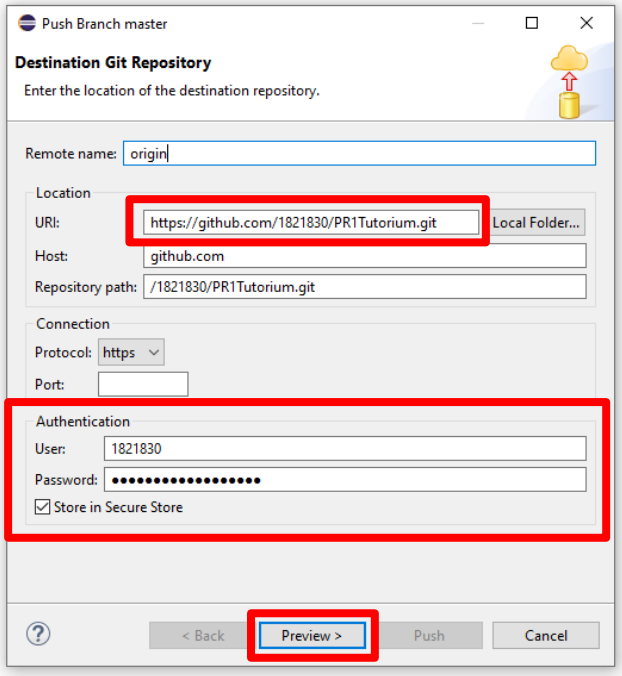
3. Kommunikation der Repositories – Commit

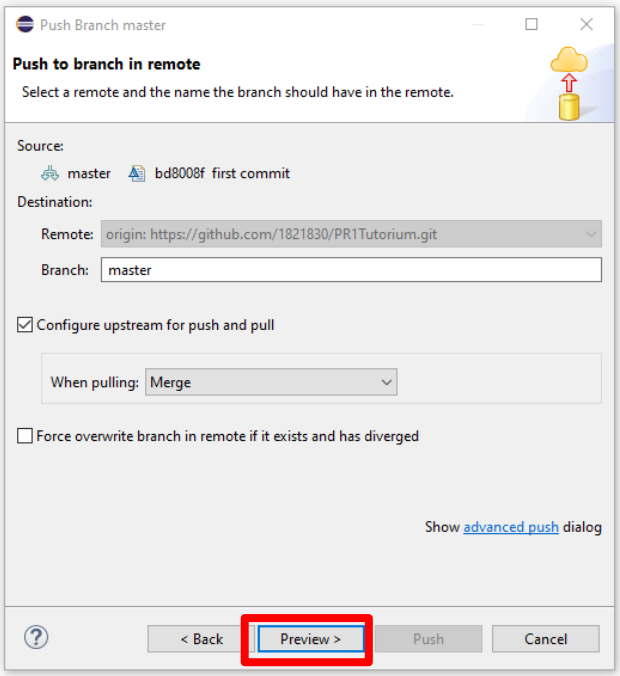


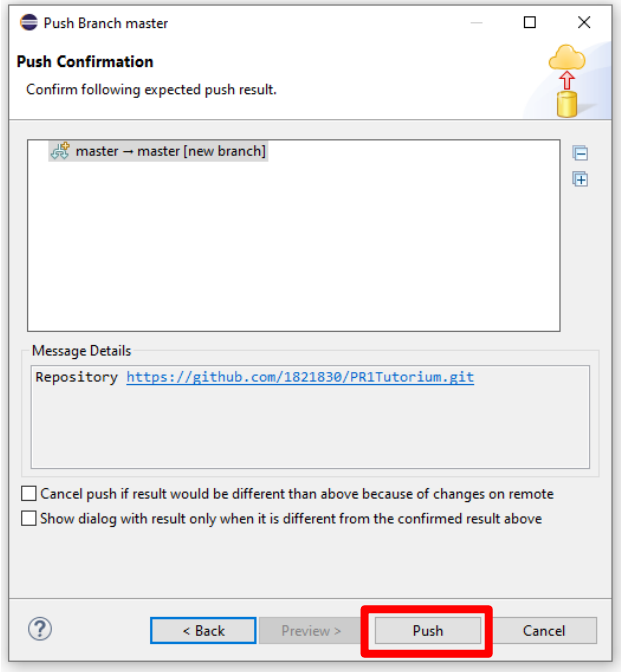
3. Kommunikation der Repositories – Push

- 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

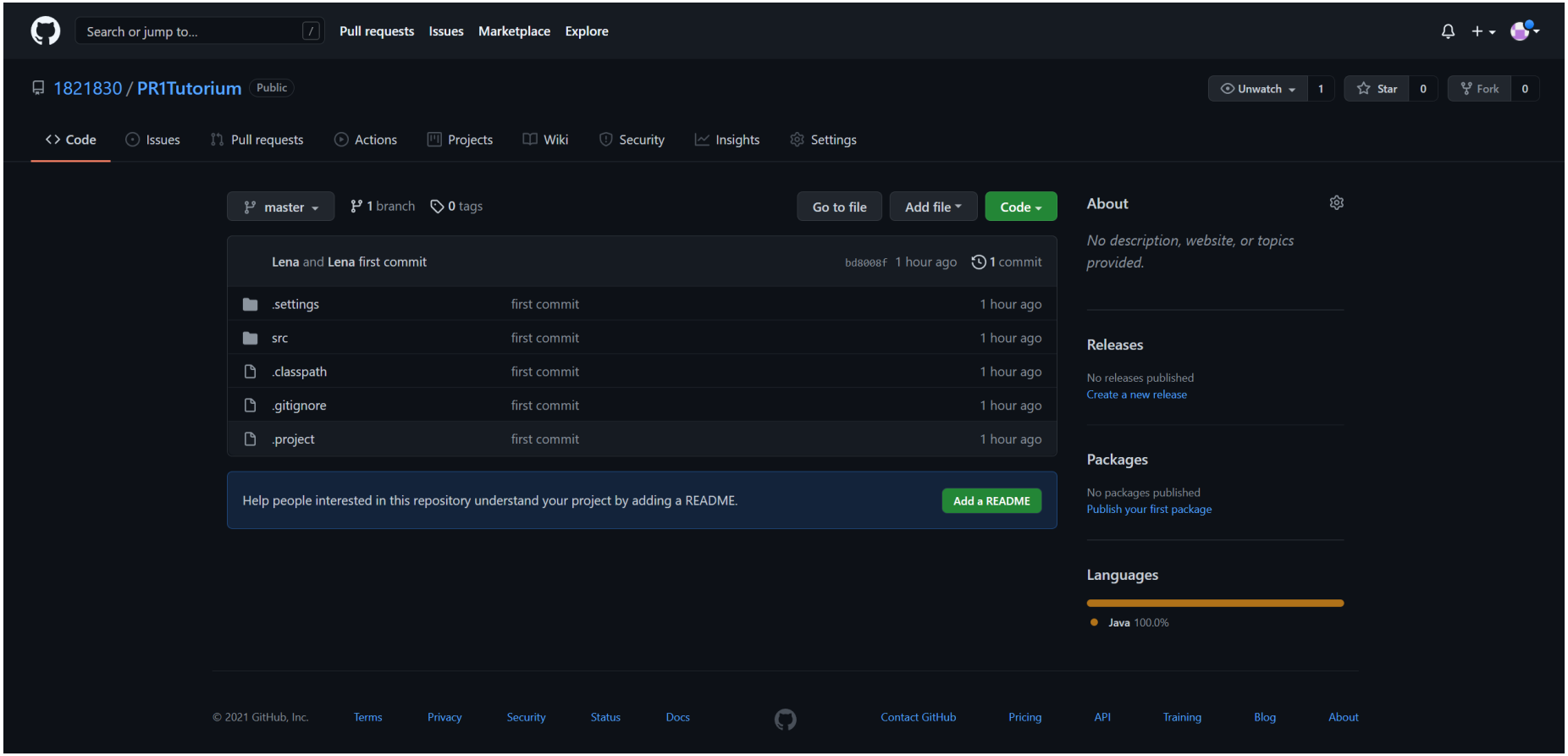
3. Kommunikation der Repositories – Push







3. Kommunikation der Repos – Push → GitHub



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)

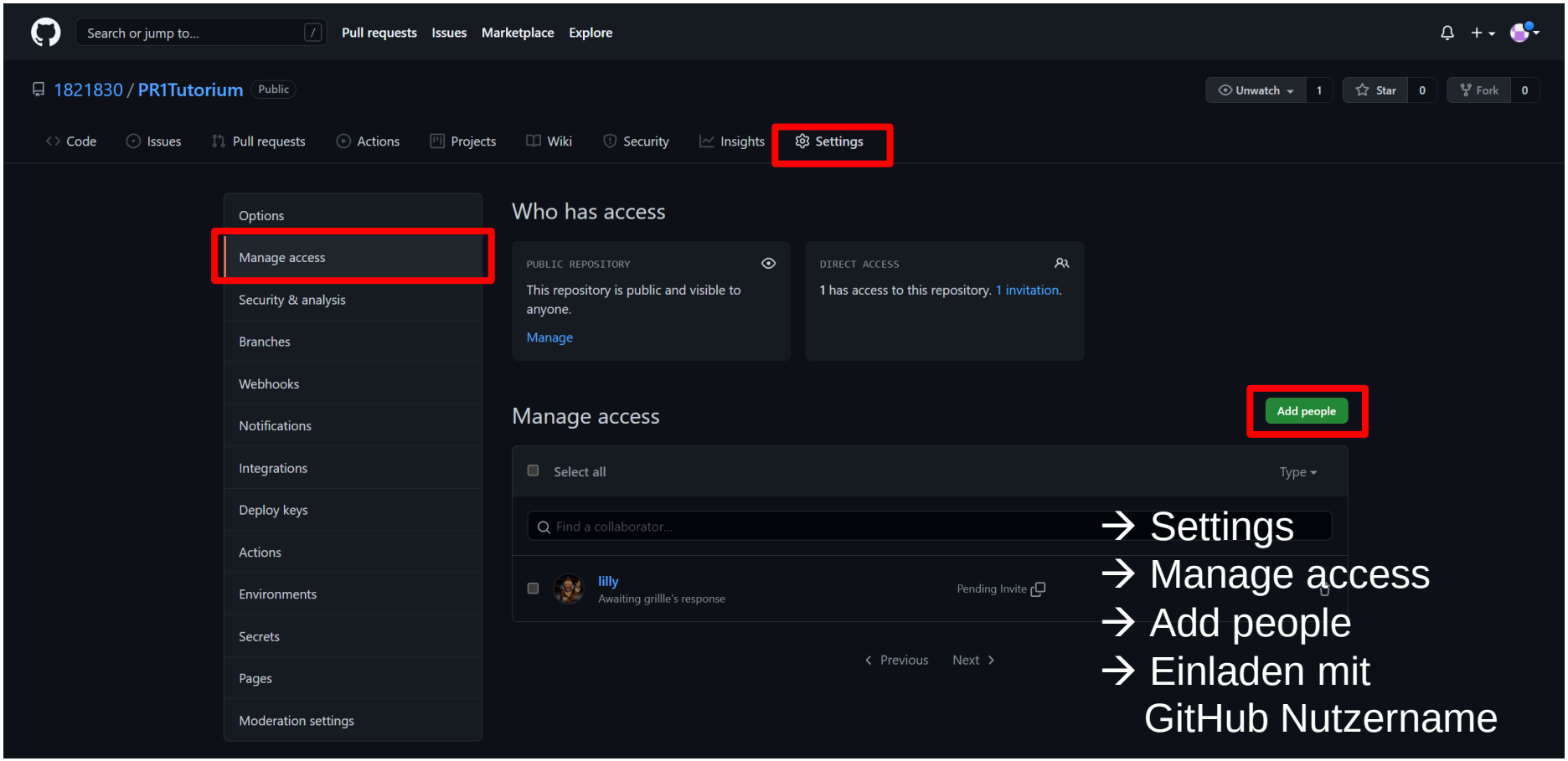
4. Verbinden / Klonen

- Das lokale und das online Repository verbinden sich über Eclipse beim ersten Puh automatisch

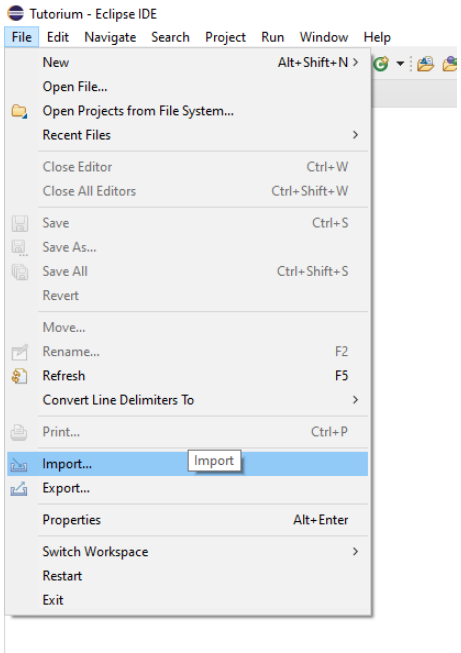
- Ein weiterer Entwickler will Zugriff auf eigenes Projekt
  → Projekt muss geklont werden

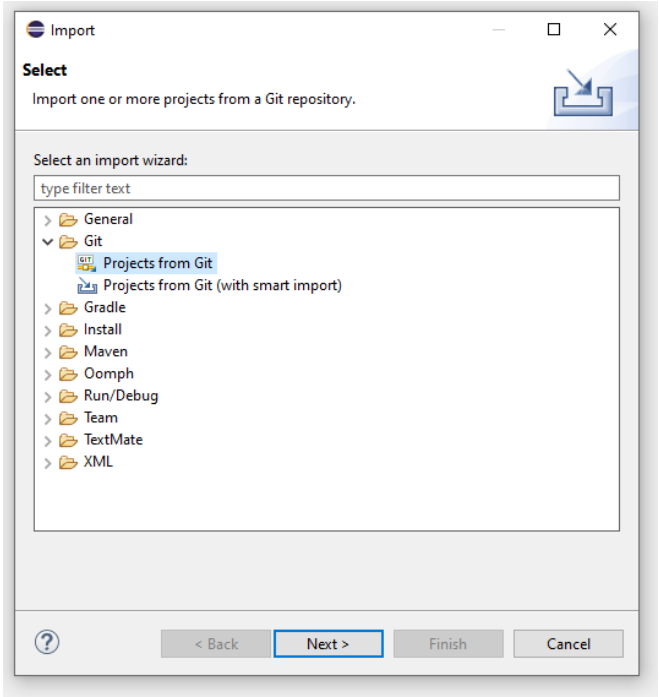
- Aktueller Stand des online Repository wird vollständig geklont
  → Es kann direkt weiter gearbeitet werden

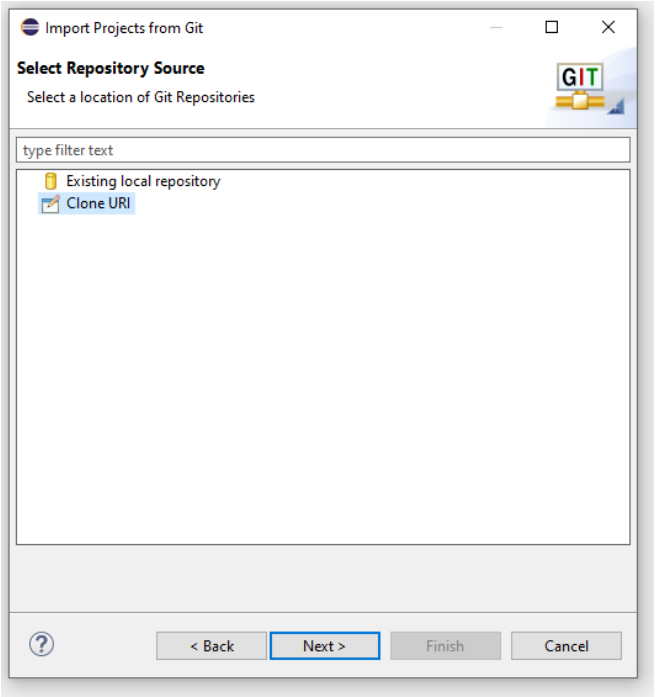
4. Klonen – Einladung in das Repository



4. Klonen in Eclipse

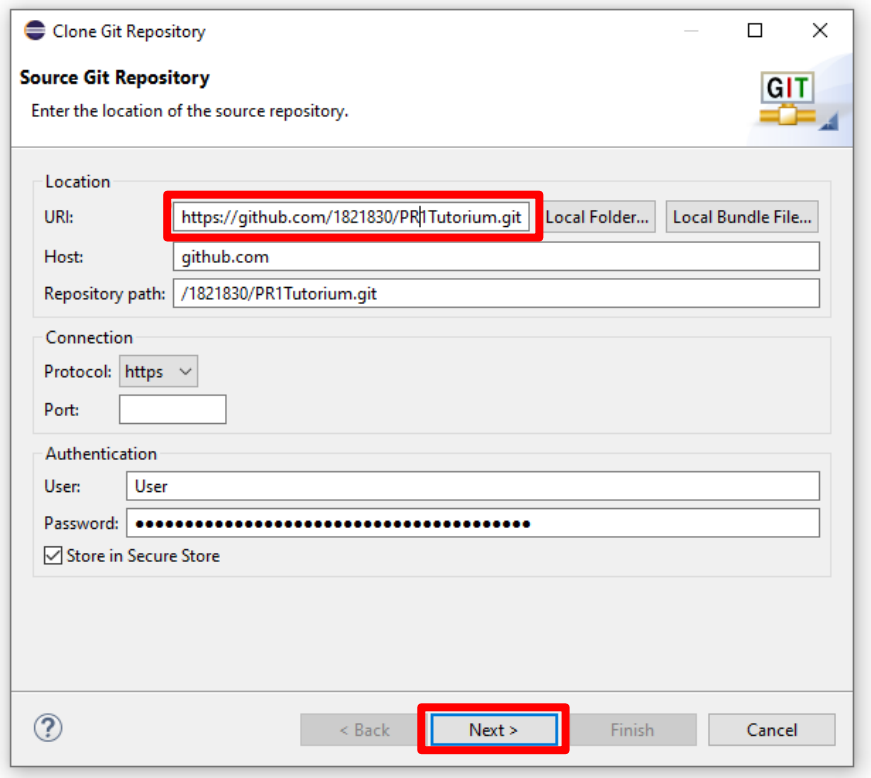


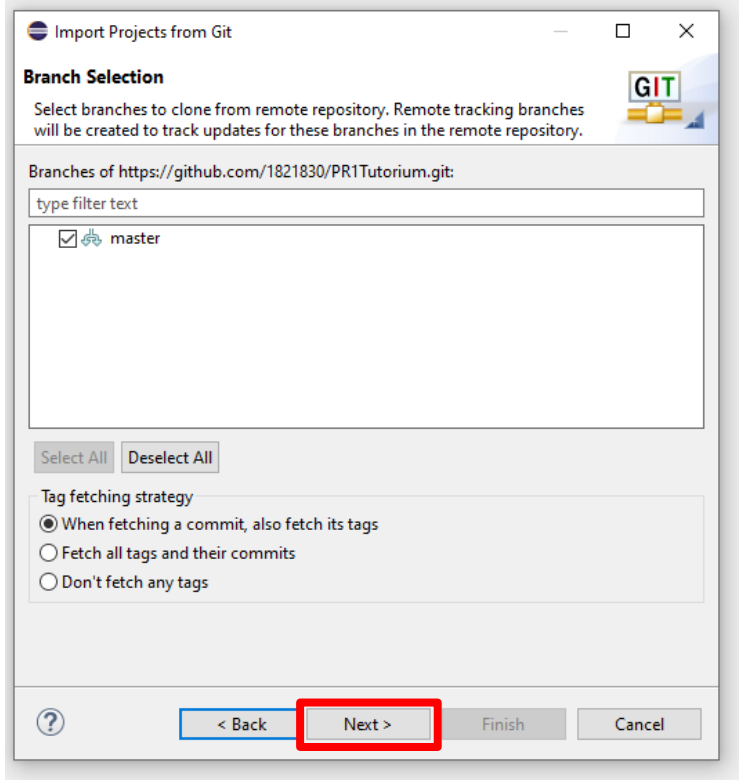




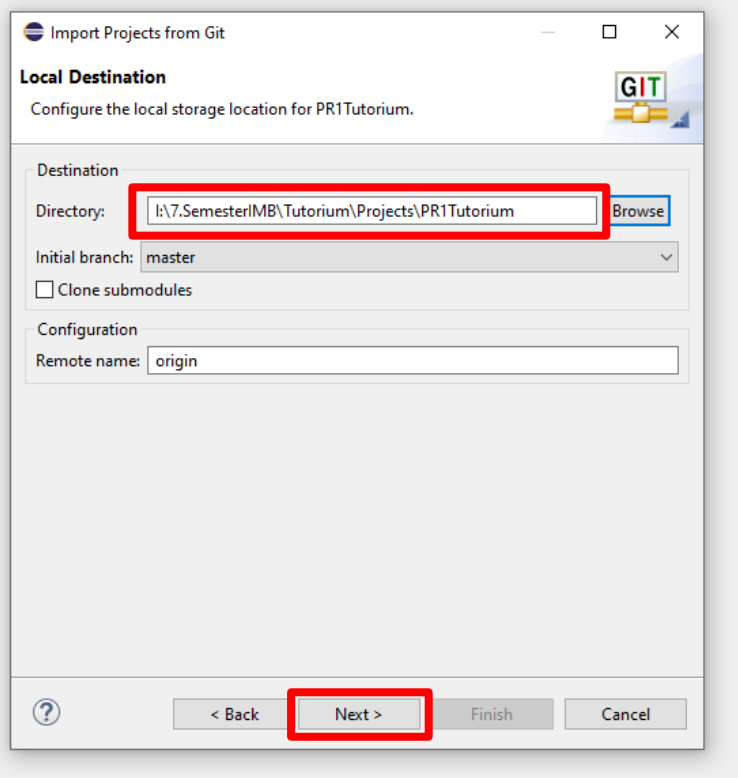
→ File → Import…
→ Projects from Git
→ Clone URI

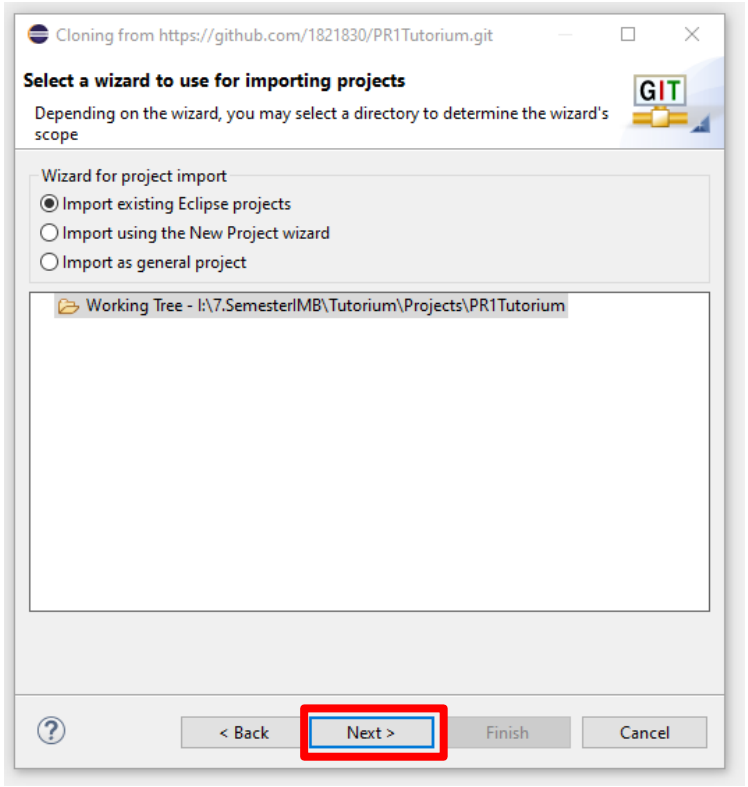
4. Klonen in Eclipse



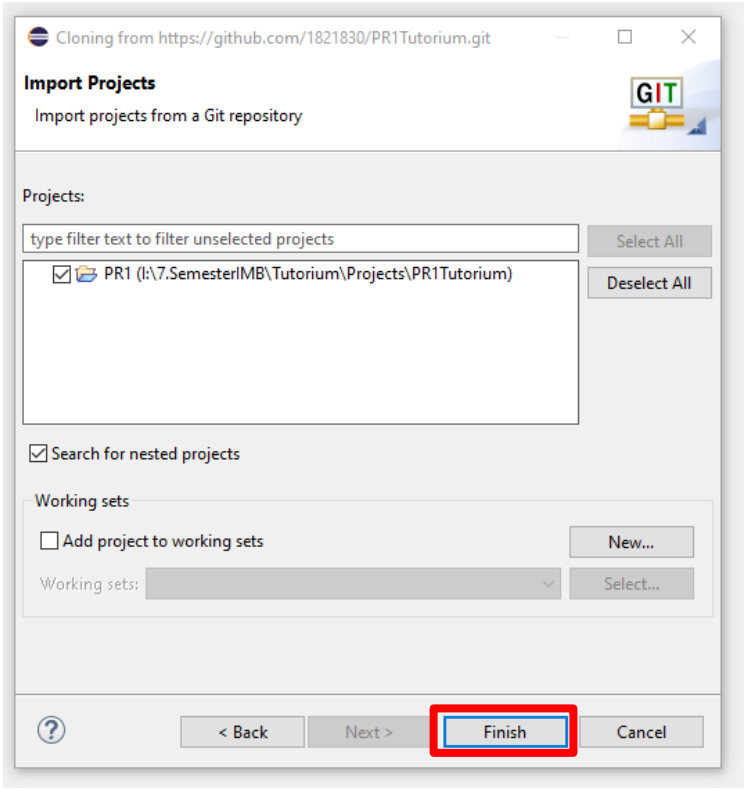


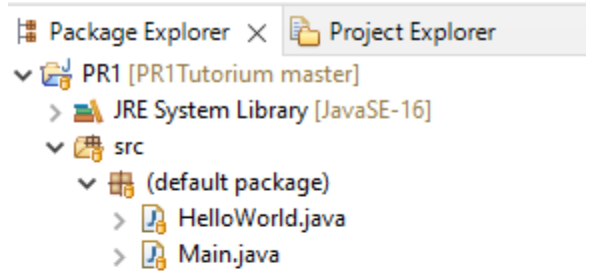
4. Klonen in Eclipse





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