Informatik
Hauptcampus

H O C H
S C H U L E
T R I E R

Anleitung zur Ausführung der Programmbeispiele aus „Parallele und verteilte Anwendungen in Java“ (6. Auflage)

Patrick Weber

August 2024

Inhaltsverzeichnis

1

Zuordnung der Submodule zu den Programmbeispielen

Die Programmbeispiele sind in mehreren Maven-Submodulen organisiert, die zu einem Elternmodul namens „puva“ gehören. Die Tabelle in Abbildung 1.1 zeigt, welche Programmbeispiele in welchem Submodul enthalten sind.

Kapitel im Buch		**Maven-Submodul**
2. Grundlegende Synchronisationskonzepte in Java		puva.concurrent
3. Fortgeschrittene Synchronisationskonzepte in Java		puva.concurrent
4. Parallelität und grafische Benutzeroberflächen		puva.jfx
5. Verteilte Anwendungen mit Sockets		puva.sockets
6. Verteilte Anwendungen mit RMI		puva.rmi
7. Verteilte Anwendungen mit indirekter Kommunikation		puva.mom
8. Webbasierte Anwendungen mit Servlets und JSF	Servlets	puva.servlets
	JSF (Java Server Faces)	puva.jsf
	RESTful WebServices	puva.rest
	WebSockets	puva.websocket
9. Verteilte Anwendungen in der Cloud		puva.cloud

Abbildung 1.1: Zuordnung der Submodule zu den Programmbeispielen.

2

Vorbereitungen

- Um die Beispiele kompilieren und ausführen zu können, benötigen Sie mindestens Java in der Version **21**.
- Obwohl die Ausführung auch ausschließlich über die Kommandozeile erfolgen kann, empfiehlt es sich, eine Entwicklungsumgebung zu verwenden. In dieser Anleitung wird hierfür die „Eclipse IDE for Enterprise Java and Web Developers“ verwendet. Die Installation der IDE und des Tomcat-Servers wird in der Anleitung „Einrichtung der Eclipse EE und des Tomcat-Servers“ erläutert.

  Achtung: Stellen Sie bitte sicher, dass Sie die Version **10.1.*** des Tomcat-Servers installieren.

3

Import des Projekts in Eclipse

Bitte entscheiden Sie sich, wo Ihr workspace gespeichert werden soll und entpacken das ZIP-Archiv in diesen Ordner. In dieser Anleitung wird als workspace `~/puva-workspace` verwendet. Der Pfad des entpackten Archivs lautet somit `~/puva-workspace/puva`.

Starten Sie bitte zunächst Eclipse und wählen den workspace. Gehen Sie zum Importieren der Maven-Module wie in den Abbildungen 3.1 bis 3.5 gezeigt, vor.

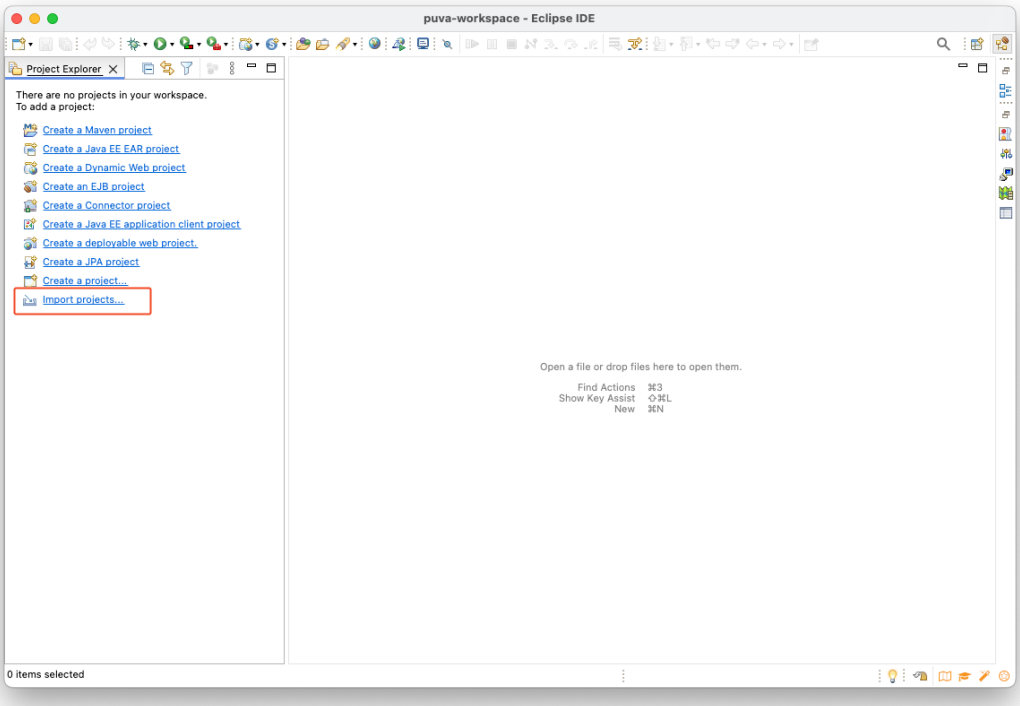

Abbildung 3.1: Import des Projekts.

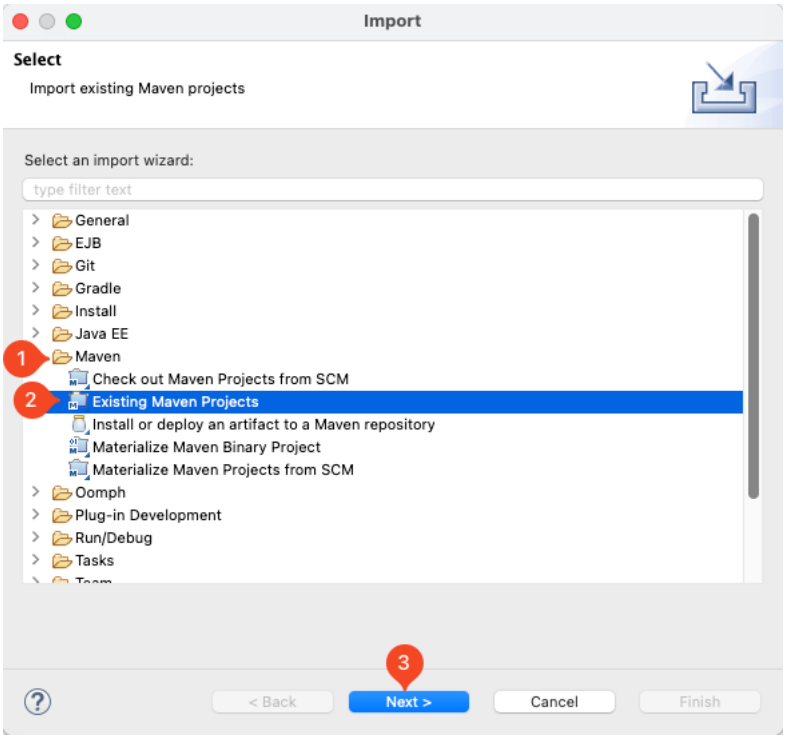

Abbildung 3.2: Import-Wizard auswählen.

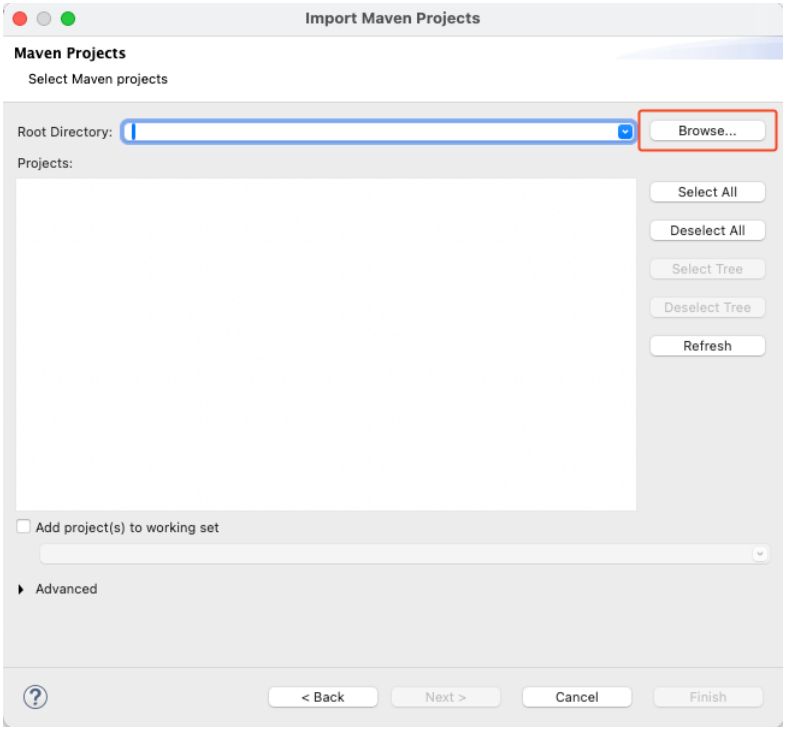

Abbildung 3.3: Wurzelverzeichnis wählen.

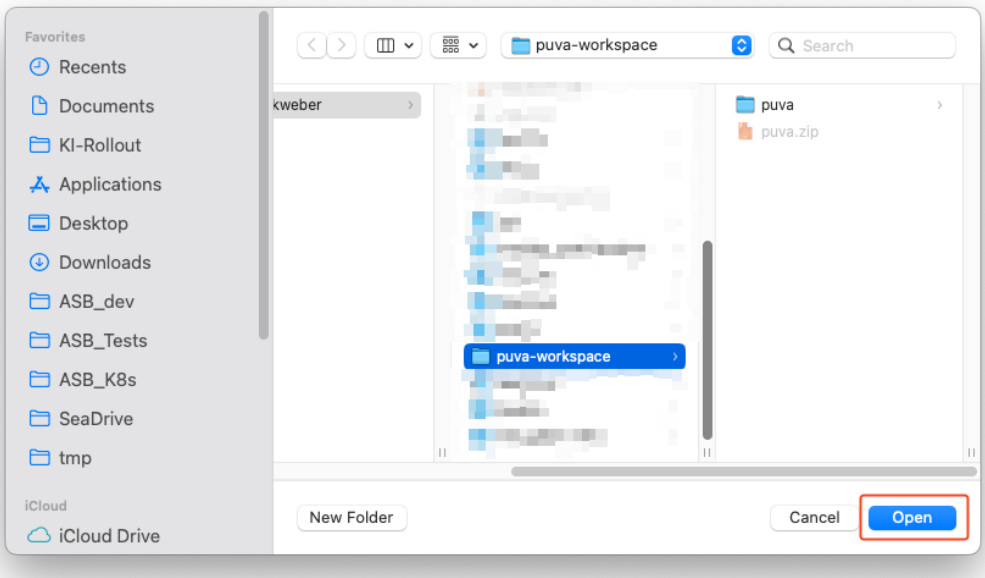

Abbildung 3.4: Zum Wurzelverzeichnis navigieren.

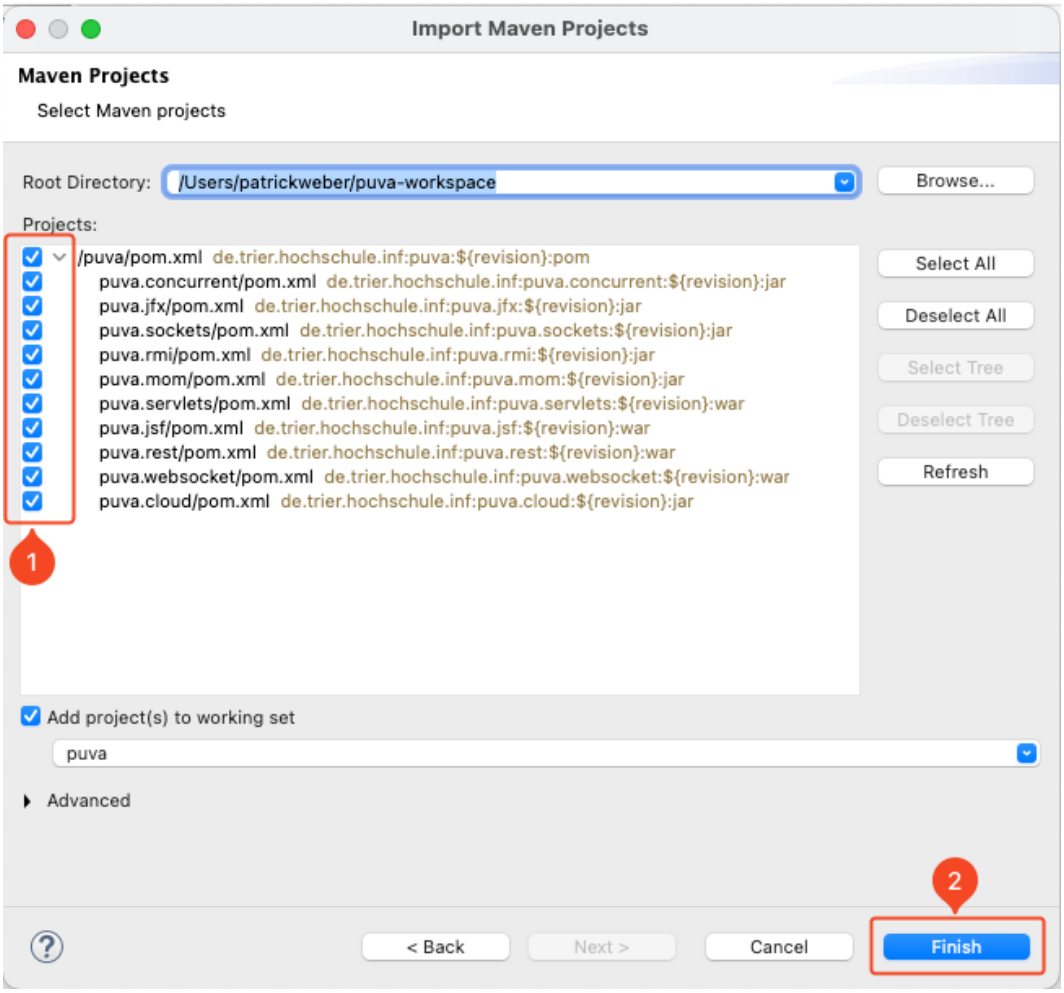

Abbildung 3.5: Sicherstellen, dass alle Module ausgewählt sind.

Nach dem Klick auf den Button mit der Aufschrift „Finish“ kann im unteren rechten Bereich der Fortschritt verfolgt werden (siehe Abbildung 3.6).

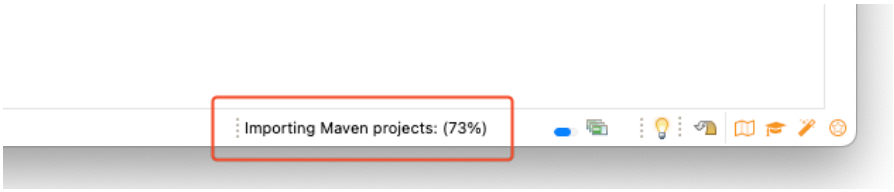

Abbildung 3.6: Fortschritt des Imports verfolgen.

Nachdem dieser Vorgang abgeschlossen ist, sollten Sie die in Abbildung 3.7 dargestellten Maven-Module im Project Explorer sehen können.

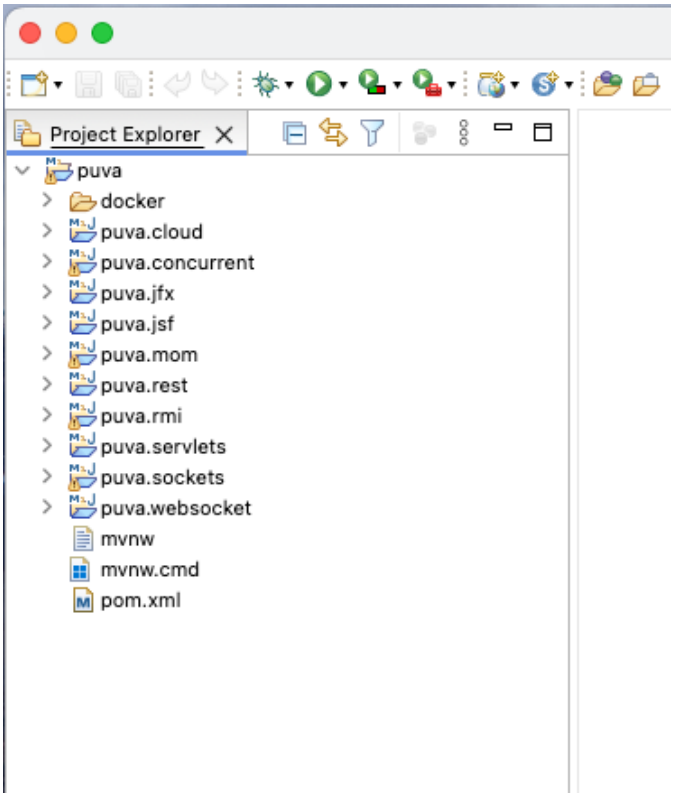

Abbildung 3.7: Project Explorer nach erfolgreichem Import.

4

Ausführung der Anwendungen über Eclipse

4.1 Beispiele aus dem Kapitel 7 des Buches

Die Beispielprogramme aus Kapitel 7 des Buches verwenden *Apache ActiveMQ Artemis*[1]. Falls Sie dieses System nicht nativ auf Ihrem Rechner installieren möchten, empfiehlt sich die Installation von *Docker*[2].

Wird Docker auf Ihrem Rechner ausgeführt, kann der Server über den folgenden Befehl gestartet werden.

```
docker run -it --rm -p 8161:8161 -p 61616:61616 apache/activemq-artemis:2.36.0
```

Im Anschluss ist er unter `http://localhost:8161/console` erreichbar. Ohne weitere Konfiguration kann zum Login der Benutzer „artemis“ mit dem Passwort „artemis“ verwendet werden.

4.2 Beispiele aus dem Kapitel 8

> Die folgenden Aktionen setzen voraus, dass Sie den Tomcat-Server in der Version **10.1.*** gemäß der Anleitung „Einrichtung der Eclipse EE und des Tomcat-Servers“ eingerichtet haben.

Zur Installation der Webapps über Eclipse, führen Sie bitte die Schritte aus, die in den Abbildungen 4.1 bis 4.3 gezeigt sind.

[1] `https://activemq.apache.org/components/artemis/`
[2] `https://www.docker.com/`

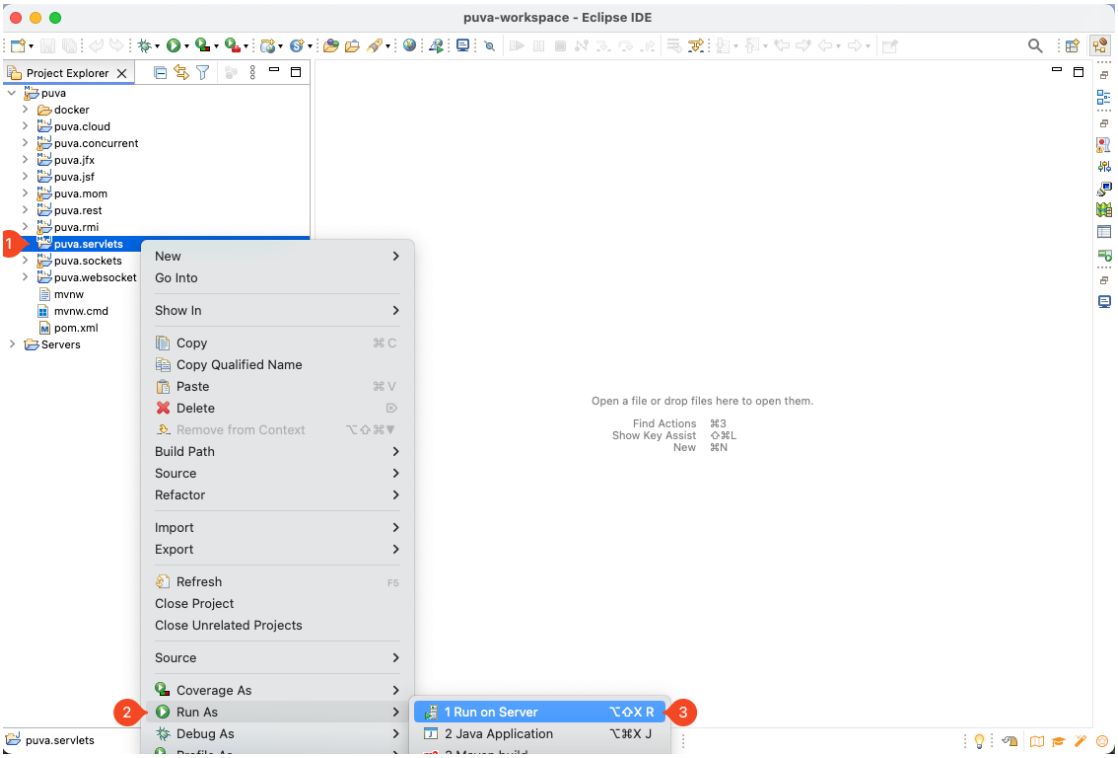

Abbildung 4.1: Webapps auf Tomcat-Server installieren

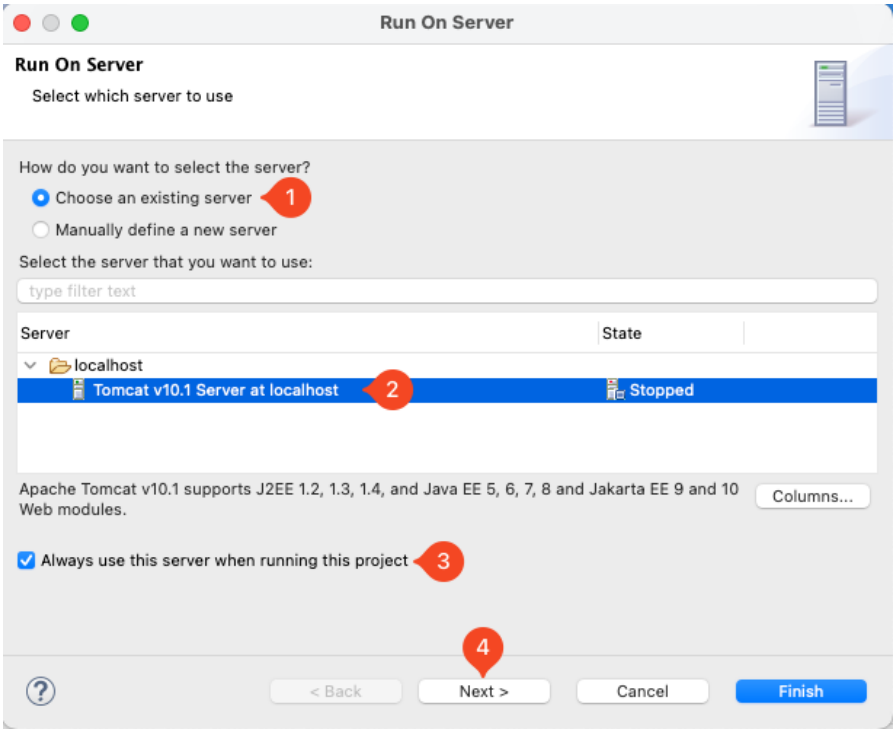

Abbildung 4.2: Tomcat-Server auswählen

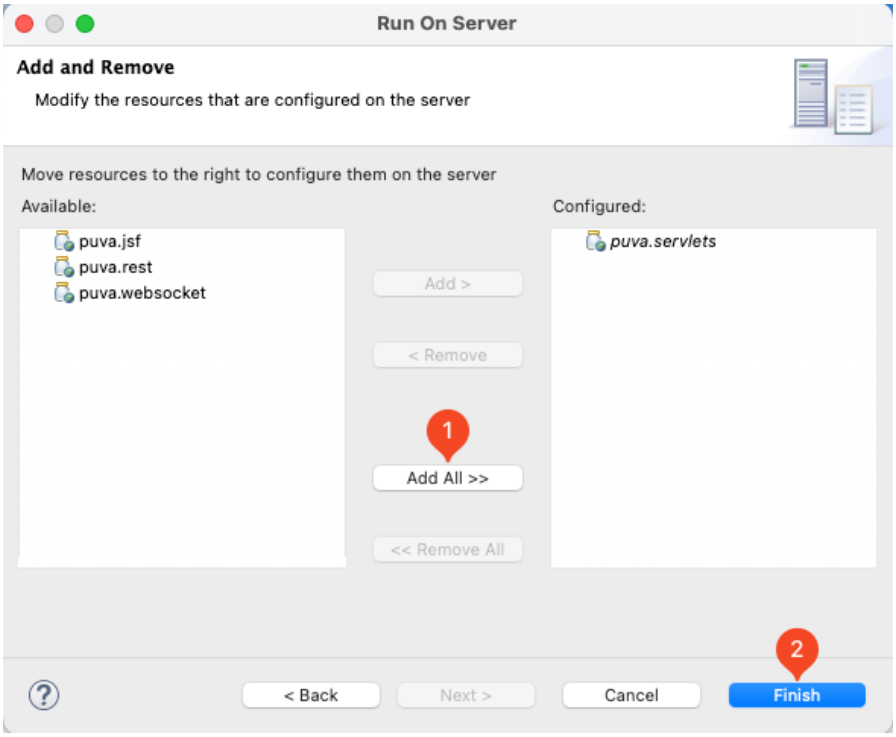

Abbildung 4.3: Alle Ressourcen auswählen

Daraufhin öffnet sich eine HTML-Seite, die auf einige Servlets verweist (siehe Abbildung 4.4). Das `HalloWelt`-Servlet ist beispielsweise unter `http://localhost:8080/puva.servlets/HalloWelt` erreichbar.

> Standardmäßig wird die HTML-Seite in Ihrem Browser geöffnet. Soll stattdessen, wie hier, der interne Browser von Eclipse verwendet werden, lässt sich das über „Preferences“ → „General“ → „Web Browser“ einstellen.

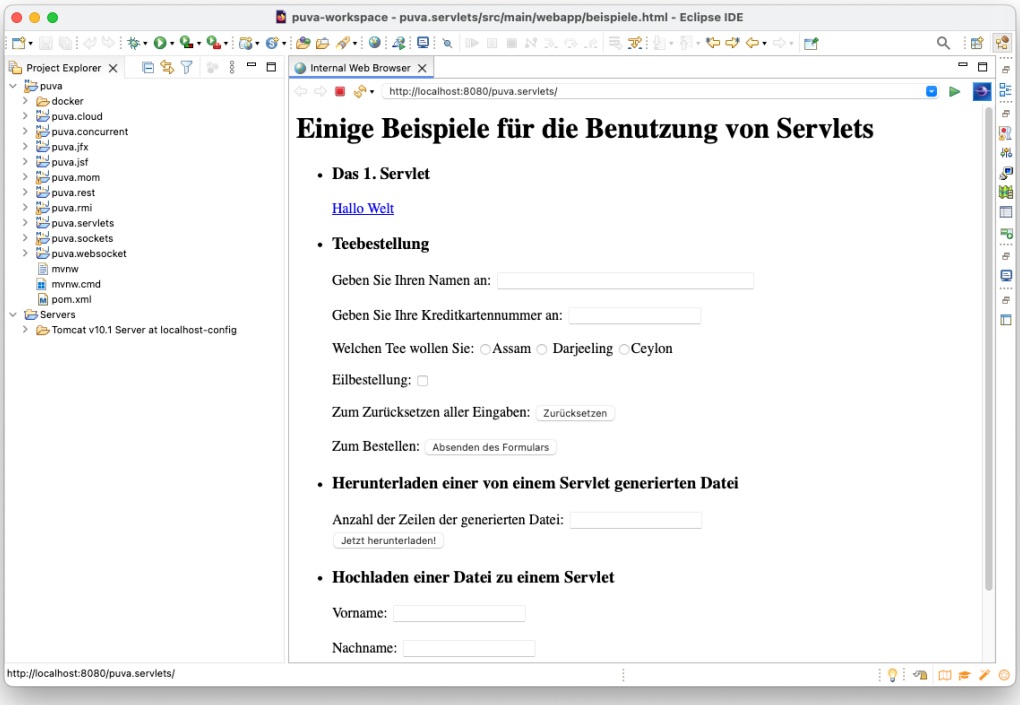

Abbildung 4.4: Beispielseite „beispiele.html“.

5

Ausführung der Anwendungen über die Kommandozeile

Falls die Beispiele über die Kommandozeile ausgeführt werden sollen, kann hierfür der Maven-Wrapper verwendet werden. Auf diese Weise ist nur die Installation von Java 21 erforderlich.

Um ein bestimmtes Ziel (engl. goal) über Maven auszuführen, kann im Wurzelverzeichnis `~/puva-workspace/puva` das Skript `mvnw` beziehungsweise unter Windows `mvnw.cmd` gefolgt von den Zielen ausgeführt werden. Standardmäßig wird das spezifizierte Ziel für jedes Submodul ausgeführt. Der folgende Befehl bewirkt beispielsweise, dass sämtliche Beispiele kompiliert und im lokalen Repository Ihres Rechners installiert werden. In Abbildung 5.1 ist die Konsole dargestellt, die zeigt, dass die Installation erfolgreich war.

```
./mvnw install
# oder unter Windows
mvnw.cmd install
```

Führen Sie ein Ziel über Maven aus, wird zunächst geprüft, ob die benötigten Abhängigkeiten (engl. dependencies) im lokalen Repository auf Ihrem Rechner vorhanden sind. Ist dies nicht der Fall, werden Sie über das zentrale Maven-Repository[1] heruntergeladen. Wie bereits erwähnt, bewirkt die Ausführung des Ziels `install`, dass ein Modul im lokalen Repository installiert wird. Hierdurch ist somit sichergestellt, dass etwaig deklarierte Abhängigkeiten zu diesem Modul erfüllt werden können.

Der oben gezeigte Befehl bewirkt die Ausführung des Ziels für alle Submodule. Soll es hingegen nur für ein bestimmtes ausgeführt werden, muss vorher in das Verzeichnis dieses Moduls gewechselt werden. Darüber hinaus muss der Pfad zum Skript `mvnw` beziehungsweise `mvnw.cmd` angepasst werden. Der folgende Befehl kann beispielsweise im Verzeichnis `~/puva-workspace/puva/puva.concurrent` abgesetzt werden und wird lediglich für das Modul „puva.concurrent“ ausgeführt.

```
../mvnw install
# oder unter Windows
../mvnw.cmd install
```

[1] `https://repo1.maven.org/maven2/`

```
[INFO] Reactor Summary for puva 6.0:
[INFO]
[INFO] puva ..................................... SUCCESS [  1.081 s]
[INFO] puva.concurrent .......................... SUCCESS [  7.157 s]
[INFO] puva.jfx ................................. SUCCESS [  1.753 s]
[INFO] puva.sockets ............................. SUCCESS [  0.264 s]
[INFO] puva.rmi ................................. SUCCESS [  0.321 s]
[INFO] puva.mom ................................. SUCCESS [ 14.856 s]
[INFO] puva.servlets ............................ SUCCESS [  2.827 s]
[INFO] puva.jsf ................................. SUCCESS [  2.683 s]
[INFO] puva.rest ................................ SUCCESS [  2.339 s]
[INFO] puva.websocket ........................... SUCCESS [  0.235 s]
[INFO] puva.cloud ............................... SUCCESS [  8.796 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

Abbildung 5.1: Erfolgreiche Installation prüfen.

Eine Java-Anwendung lässt sich über ein spezielles Maven-Plugin ausführen, wobei die Hauptklasse über die Property `exec.mainClass` anzugeben ist. Hierfür ist das Ziel `exec:java` nach der Kompilierung auszuführen. Es gilt zu beachten, dass dieses Ziel nur in den Submodulen ausgeführt werden kann. Der folgende Befehl funktioniert mit dieser Hauptklasse daher nur im Verzeichnis `~/puva-workspace/puva/puva.concurrent`.

```
../mvnw compile exec:java -Dexec.mainClass=chapter2.Banking
```

Programmargumente können über die Property `exec.args` übergeben werden:

```
../mvnw compile exec:java -Dexec.mainClass=chapter5.tcp.TCPSleepClient -Dexec.
    ↪ args="localhost 4"
```

Analog dazu können Java-FX-Anwendungen ausgeführt werden. Es ändert sich lediglich das auszuführende Ziel:

```
../mvnw javafx:run -Dexec.mainClass=chapter4.basics.DrawingLines
```

Das Ziel `compile` muss hier nicht explizit angegeben werden, da es durch `javafx:run` automatisch ausgeführt wird.

Alternativ kann die Hauptklasse der auszuführenden Java-FX-Anwendung auch in der Datei „pom.xml“ des Submoduls konfiguriert werden. Soll zum Beispiel das „HelloWorld“-Programm aus Kapitel 4 des Buches ausgeführt werden, muss die folgende Zeile in der Datei `~/puva-workspace/puva/puva.jfx/pom.xml` einkommentiert werden:

```
<mainClass>chapter4.basics.HelloWorld</mainClass>
```

Beachten Sie bitte, dass nur das zuletzt gefundene mainClass-Element berücksichtigt wird. Möchten Sie die auszuführende Anwendung ändern, kommentieren Sie die anderen mainClass-Elemente am besten aus.

5.1 Beispiele aus dem Kapitel 8 des Buches mit Docker

Falls Sie den Tomcat-Server in einem Docker-Container ausführen möchten, ist die Bereitstellung (engl. deployment) der Webapps über ein Maven-Plugin am einfachsten. Hierfür ist zunächst ein Docker-Image zu erzeugen, in dem die *Tomcat Manager App* konfiguriert wird. Das hierfür benötigte Dockerfile befindet sich im Verzeichnis `~/puva-workspace/puva/docker`. Die folgenden Befehle können in diesem Verzeichnis abgesetzt werden, um das Image zu erzeugen und einen Container daraus zu starten.

```
docker build --tag tomcat-puva:10.1.28 .
docker run -it --rm -p 8080:8080 tomcat-puva:10.1.28
```

Die Installation der Webapps erfolgt über die Ausführung des Ziels `tomcat7:deploy`. Sie können es auf dem Eltern- oder Submodul ausführen, um alle Webapps, beziehungsweise nur die des Submoduls zu installieren. Entscheiden Sie sich beispielsweise für das Submodul „puva.servlets“ (oder das Elternmodul) und führen Docker auf Ihrem Rechner aus, ist das `HalloWelt`-Servlet im Anschluss unter `http://localhost:8080/puva.servlets/HalloWelt` erreichbar.