Open Liberty: Installation und Konfiguration des Application Servers

Vorbemerkungen

- Betriebssysteme und zugehörige Eigenschaften

  Open Liberty steht für alle in unseren Seminaren genutzten Betriebssysteme (Windows, Linux, Mac OS) zur Verfügung.

  Wir geben in diesem Dokument Dateinamen im Windows-Format an. Für Linux und Mac OS ersetzen Sie bitte das Trennzeichen `\` durch `/`.

  Kommandoskripte sind für Windows i. d. R. `.bat`-Dateien (teilweise sind auch `.cmd` oder `.ps1` vorhanden). Unter Linux und Mac OS nehmen Sie stattdessen `.sh`-Dateien.

- Server-Bereitstellung innerhalb der Seminarprojekte

  Die meisten unserer Seminarprojekte im JEE- und MicroProfile-Bereich sind so konfiguriert, dass ein Liberty-Server darin automatisch bereitgestellt und zum Projekt passend konfiguriert werden kann. Details dazu finden Sie in den Projektverzeichnissen jeweils in der Datei `readme.adoc`. Ihre TrainerIn unterstützt Sie natürlich bei der Nutzung.

  Die im Projekt bereitgestellten Server haben den Vorteil, dass sie einen sog. *Development Mode* unterstützen, in dem die Seminaranwendung automatisch deployt und nach Änderungen synchroniert wird. Zudem wird keinerlei Integration in die genutzte IDE erforderlich.

  Die nachfolgend beschriebenen Punkte entfallen dann mit Ausnahme von *Konfiguration des Logging-Systems*.

Installation und Konfiguration des Servers

> **TIP** Bei Seminaren, die Open Liberty benötigen, werden die hier beschriebenen Schritte (Download, Installation, Erzeugung einer Konfiguration für das Seminar, Einrichten von Ressourcen) durch den Aufruf von `mvn` im Verzeichnis `labs` bereits durchgeführt. Der Server steht Ihnen im Verzeichnis `labs\tools\target\openliberty-22.0.0.1\wlp` zur Verfügung. Das Unterverzeichnis `usr\servers\seminar` enthält die für das Seminar angepasste Serverkonfiguration.

Download und Installation

Open Liberty kann von `https://openliberty.io/downloads/` heruntergeladen werden. Im Seminar

wird die Version *22.0.0.1* genutzt.

Das heruntergeladene File `openliberty-22.0.0.1.zip` kann an beliebiger Stelle entpackt werden. Dabei entsteht ein neues Verzeichnis namens `wlp`, das im Rest dieses Dokumentes mit `<wlp_home>` bezeichnet wird.

Erzeugung einer Server-Konfiguration für das Seminar

Öffnen Sie ein Kommandofenster (bzw. Shell) mit dem aktuellen Verzeichnis `<wlp_home>\bin` und starten Sie dort das Kommando `server create seminar`. Damit wird eine Serverkonfiguration im Verzeichnis `<wlp_home>\usr\servers\seminar` erzeugt.

Kopieren Sie die Dateien aus `tools\setup\openliberty` nach `<wlp_home>`. Dadurch werden u. a. die folgenden Einstellungen im Server (genauer: in der Datei `usr\servers\seminar\server.xml`) gemacht:

- Auswahl der Java-EE-Features (Element `<feature>`).
- Ändern der Ports für *HTTP* und *HTTPS* auf `8080` und `8443` statt der voreingestellten `9080` und `9443` (Element `<httpEndpoint>`).
- Konfiguration zweier Datasources `DefaultDataSource` und `seminar` (Elemente `<dataSource>`).
- Setzen des Log-Levels für `de.gedoplan` auf `fine` (Element `<logging>`).
- Freigabe von Batch-Operationen für alle User (Element `authorization-roles`).

Zudem wird durch die Datei `etc\jvm.options` die Laufzeit-Sprache auf Englisch gesetzt.

Für die Datasources wird noch der Treiber für die referenzierte H2-Datenbank benötigt. Er liegt im `tools\target`-Verzeichnis in der Datei `h2-x.y.z.jar`. Legen Sie das Verzeichnis `<wlp_home>\usr\shared\resources\h2` and und kopieren Sie das Treiber-`.jar`-File dort hinein.

Start und Stopp des Servers

> **TIP** Im Seminar (und auch sonst zur Entwicklung von Software) ist es empfehlenswert, den Server nicht wie gezeigt separat zu starten, sondern ihn in die genutzte IDE zu integrieren und von dort zu kontrollieren.

Der Server kann mit dem Kommando `server` im Verzeichnis `<wlp_home>\bin` gestartet und gestoppt werden. Die folgenden Kommandovarianten stehen dafür zur Verfügung:

`server start seminar`
: Starten des Servers `seminar` im Hintergrund, d. h. ohne eigenes Fenster.

`server stop seminar`
: Stoppen des Servers `seminar`.

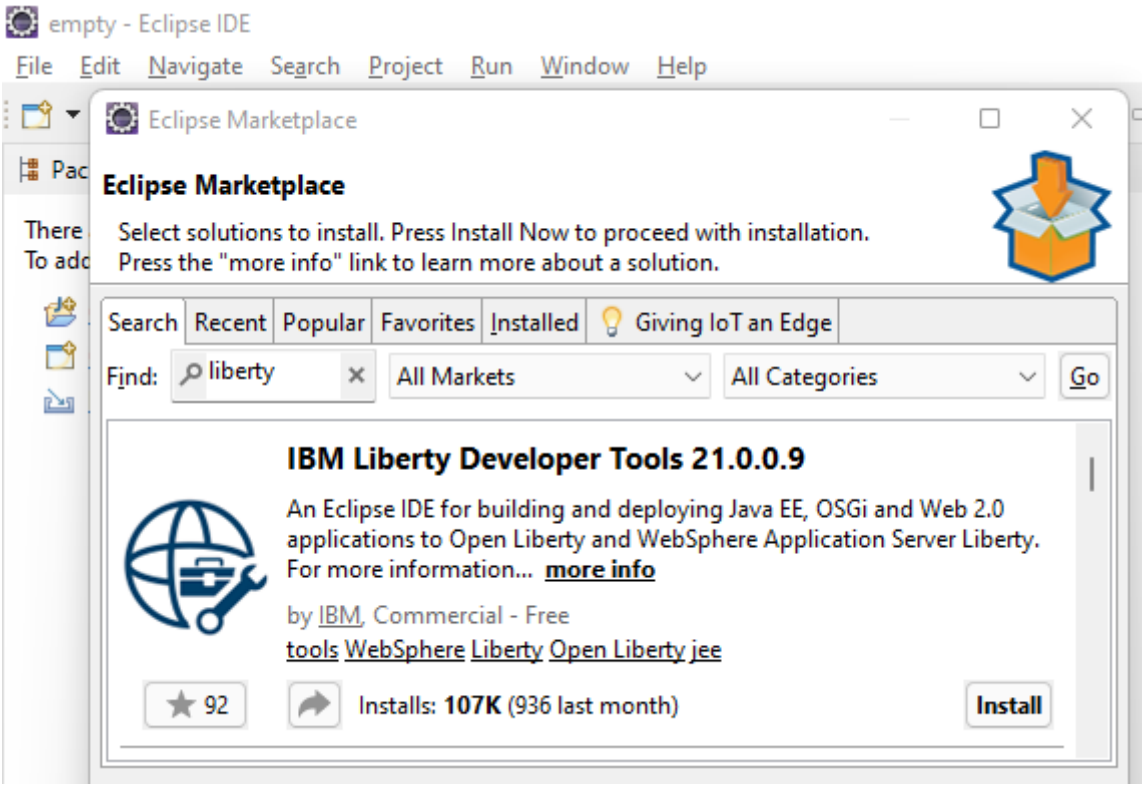
`server run seminar`
: Starten des Servers `seminar` im Vordergrund, d. h. mit Ausgabe der Protokollausgaben ins

Kommandofenster.

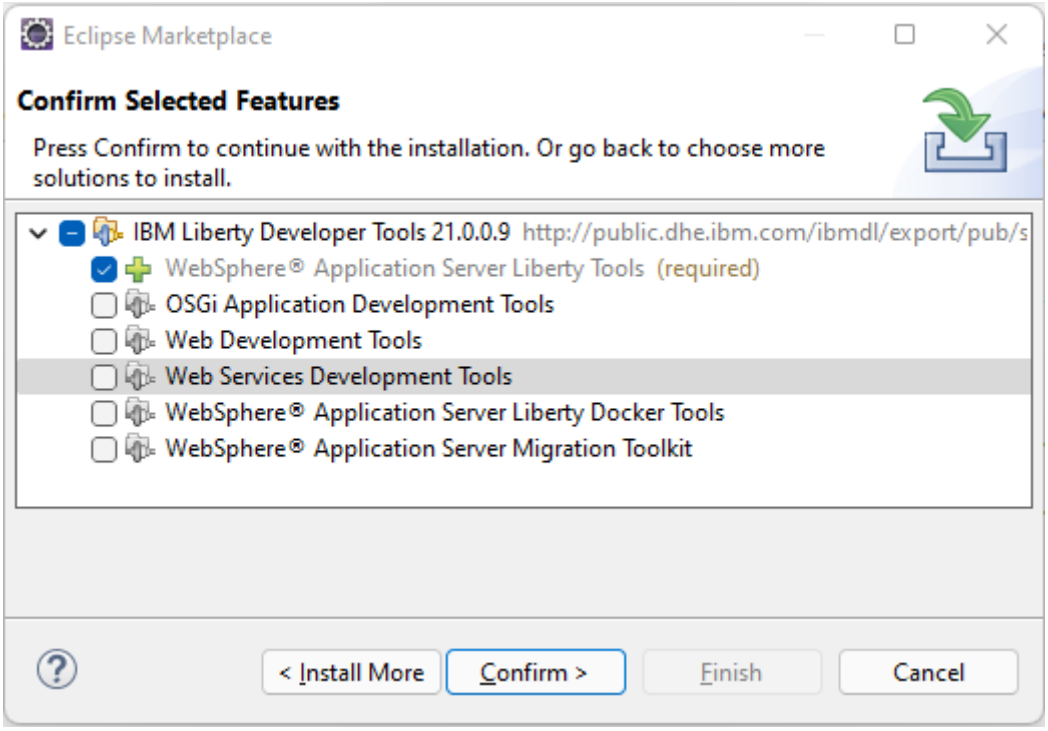
Integration des Servers in die IDE

Eclipse

- Installieren Sie zunächst die `IBM Liberty Developer Tools`. Dazu wählen Sie im Menü `Help` den Punkt `Eclipse Marketplace` und suchen nach `liberty`:



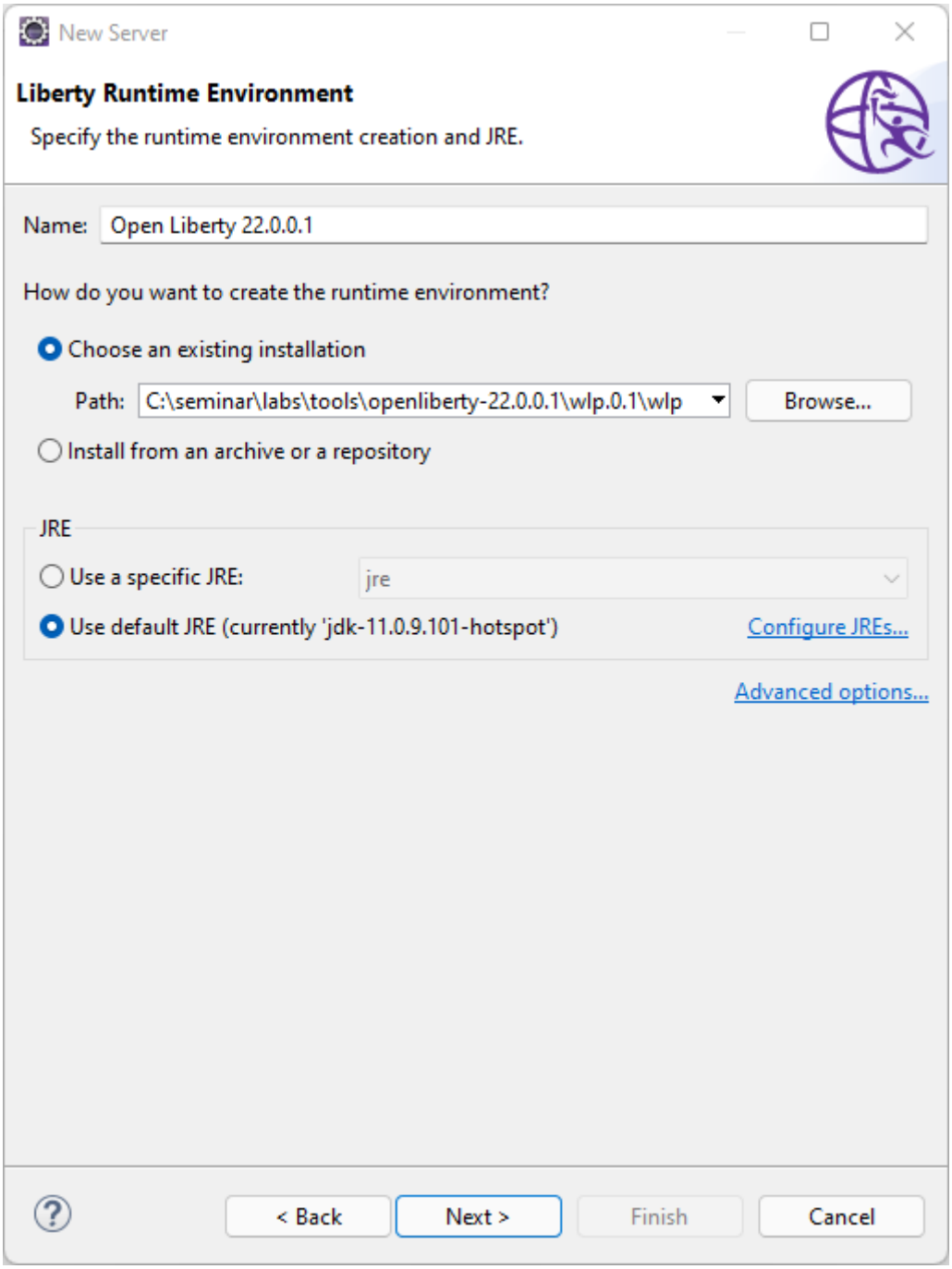
Installieren Sie die `IBM Liberty Developer Tools` mit dem Button `Install`. Im folgenden Dialog können Sie die nicht mit `(required)` gekennzeichneten Features deaktivieren:



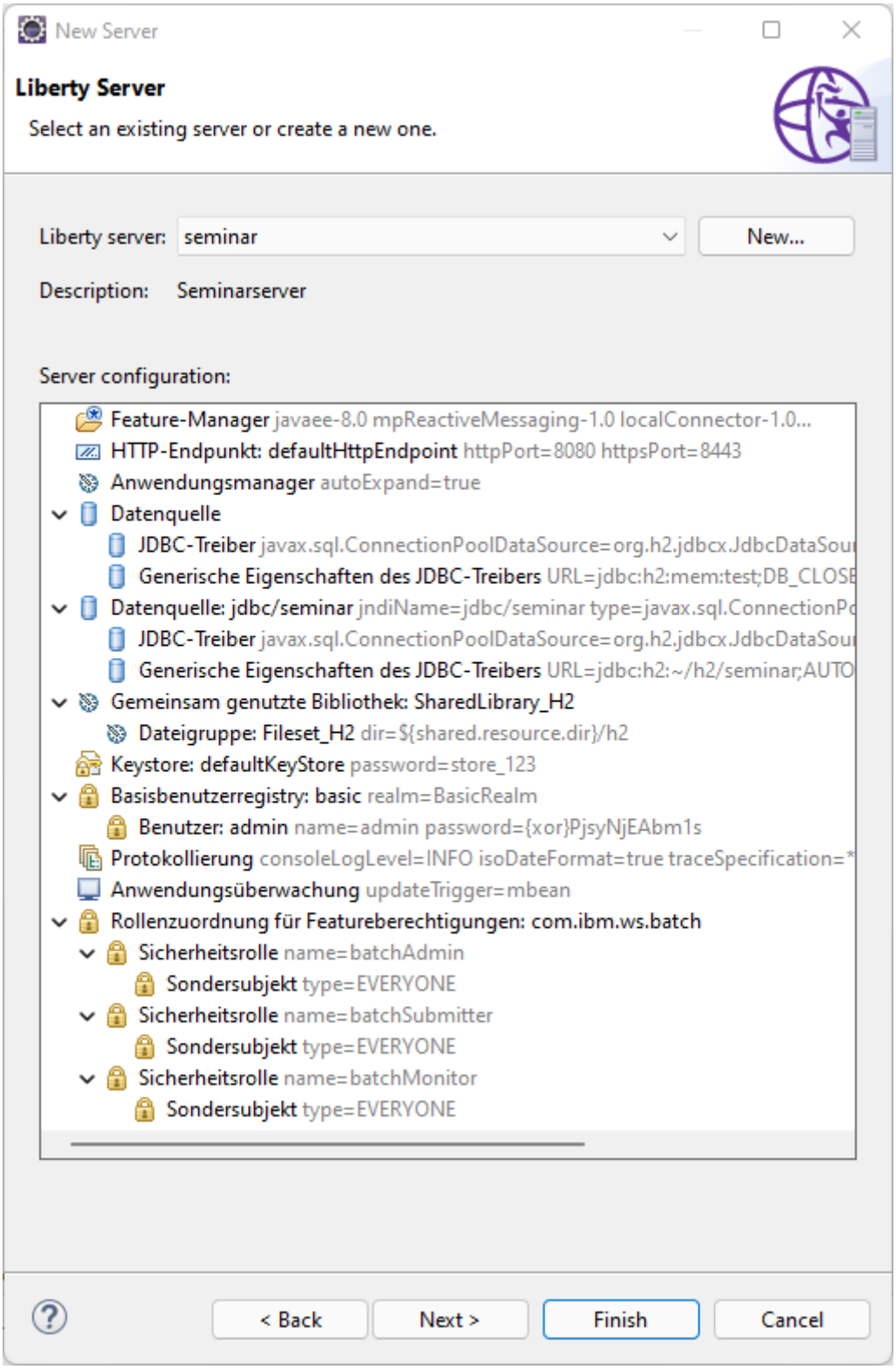
Die Installation benötigt einige Minuten und erfordert zwischendurch einen Restart der IDE.

- Fügen Sie die View `Servers` Ihrer genutzten Perspektive hinzu. Dazu nutzen Sie den Menüpunkte `Window` → `Preferences` → `Show View` → `Other···` und wählen die View namens `Servers` aus.
- Klicken Sie mit der rechten Maustaste in den freien Bereich der View `Servers` , wählen aus dem Kontextmenü `New` → `Server`, klicken aus dem Ordner `IBM` den Eintrag `Liberty Server` an und setzen als `Server name Open Liberty 22.0.0.1` ein. Auf der nächsten Dialogseite geben Sie den Pfad `<wlp_home>` an. Als `JRE` benötigen Sie ein JDK der Version 11 oder höher.

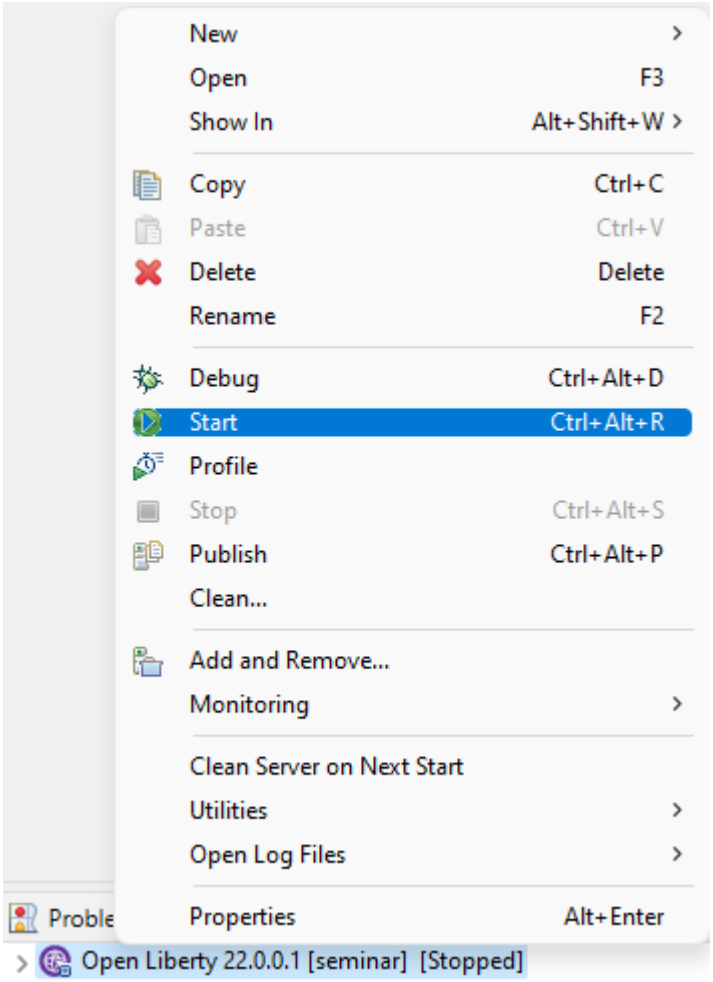
CAUTION	Das zuvor gesagte setzt voraus, dass Sie noch keinen Liberty Server in Ihrem Eclipse Workspace konfiguriert hatten. Sollte das doch der Fall sein, klicken Sie im ersten Dialog auf den Link `Add···`, um ein neues *Server runtime environment* anzulegen.



- Nach Klick auf `Next` wählen Sie den `Liberty server seminar` aus und beenden den Dialog mit `Finish`.

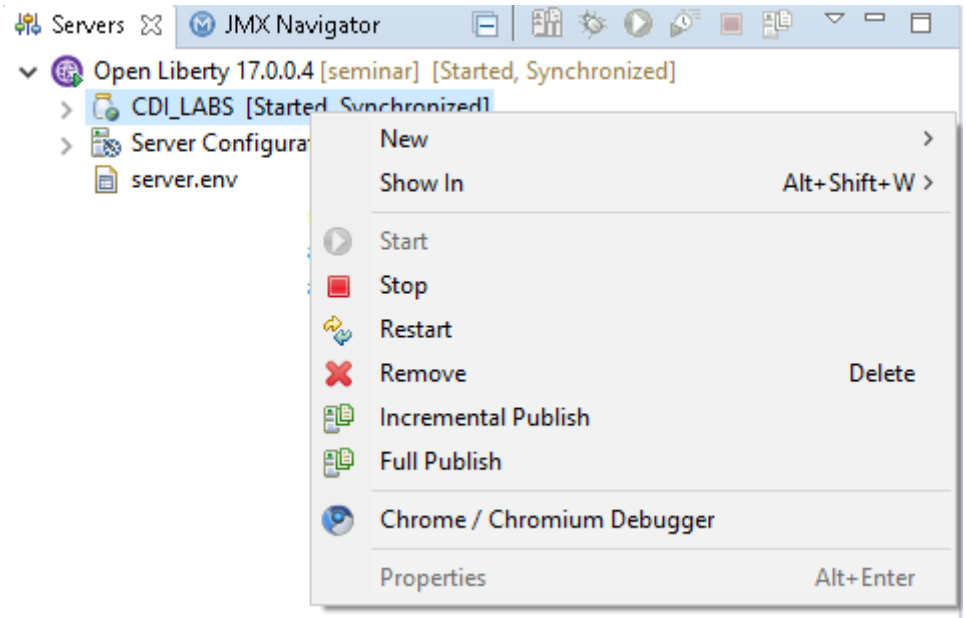


- Nach Abschluss des Konfigurationsdialogs mit `Finish` erscheint ein entsprechender Eintrag in der View `Servers`. Nach einem Rechtsklick darauf kann der Server gestartet (und später auch wieder gestoppt) werden.



Deployment von Anwendungen

Anwendungen können per Drag-and-Drop in den Server gebracht werden. Dazu ziehen Sie das gewünschte Projekt aus der View `Package Explorer` (oder `Projekt Explorer`) auf den Servereintrag in der View `Servers`. Die Anwendung erscheint dann dort eingerückt unterhalb des Servereintrags und kann mit einem Rechtsklick erneut deployt (`Full Publish`) oder wieder entfernt werden (`Remove`).



Konfiguration des Logging-Systems

In den Demo- und Übungsklassen wird *Apache Commons Logging* zur Protokollierung verwendet. Es handelt sich dabei um ein Meta-Logging-Framework, das zur Laufzeit das Log-System des Zielservers verwendet.

Open Liberty nutzt ein proprietäres System, das auf die Standardklassen aus `java.util.logging` aufbaut. Die Log-Meldungen werden im Verzeichnis <wlp_home>\usr\servers\seminar\logs` in den Dateien `console.log`, `messages.log` und `trace.log` abgelegt. Die Ausgaben in `console.log` erscheinen auch in der Standard-Ausgabe.

Für die Protokollierung aus Anwendungen heraus eignet sich das Trace Log, das durch das folgende Element in der Server-Konfigurationsdatei `server.xml` konfiguriert wird:

```
<logging
  consoleLogLevel="info"
  traceSpecification="*=info:de.gedoplan=finest">
</logging>
```

Das Attribut `traceSpecification` bestimmt dabei, welche Meldungen in `trace.log` eingetragen werden. Es enthält eine durch `:` getrennte Liste von Logger-Konfigurationen der Form `name=level`.

- `name` stellt darin üblicherweise einen Paket- oder Klassennamen dar. `*` stellt die globale Grundeinstellung dar. Für jeden Logger gilt die Einstellung, die seinen Namen am genauesten spezifiziert, d. h. ein Logger, der in der Anwendung mit dem Namen `de.gedoplan.seminar.cdi.demo.basics.presentation.DemoPresenter` erzeugt und genutzt wird, kann mit einem Konfigurationseintrag `de.gedoplan=fine` konfiguriert werden. Gibt es dagegen auch einen Eintrag `de.gedoplan.seminar.cdi.demo=finest`, so gilt dieser.
- `level` bestimmt, ob Meldungen ausgegeben oder ausgefiltert werden, z. B.:
  - `severe`: Fehlermeldungen (in anderen Log-Frameworks `ERROR`).
  - `warning`: Warnungen (in anderen Log-Frameworks `WARN`).
  - `info`: Allgemeine Infos (in anderen Log-Frameworks `INFO`).
  - `fine`: Debug-Meldungen (in anderen Log-Frameworks `DEBUG`).
  - `finest`: Trace-Meldungen (in anderen Log-Frameworks `TRACE`).

Das Attribut `consoleLogLevel` konfiguriert den Filter für `console.log`. Leider kann man hier nur Levels bis `info` eintragen, d. h. Debug- und Trace-Meldungen erscheinen dort nicht.

Änderungen an der Konfigurationsdatei `server.xml` können im laufenden Betrieb gemacht werden und werden sofort aktiv.