

Installationsanweisung der ,XXS Pumperapp'

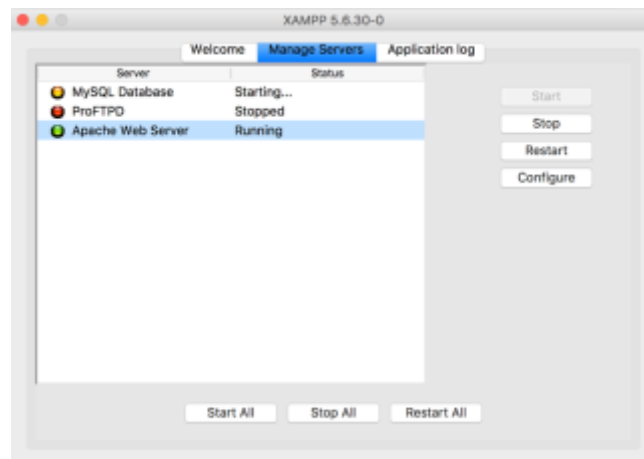
Voraussetzungen für das lokale Deployment der Applikation

Hinweis: Das Projekt wurde lediglich auf Windows und MacOS X entwickelt. Wenngleich das Deployment auch auf Linux kein Problem darstellen sollte, konnte der Prozess aus Mangel eines realen Gerätes nicht getestet werden.

1. Lokale Root-Rechte auf dem Client
2. [Xampp](#) ab der Version 7.0 für Windows, Linux oder MacOS X
3. [Eclipse](#) ab der Version 4.6 für Windows, Linux oder MacOS X
4. [Maven](#) ab der Version 3.3
5. [Eclipse-Maven-Plugin](#)
6. [Vaadin Pro-Lizenz](#). Lokal gespeicherter Vaadin Pro-Key für [Vaadin Charts](#)
7. Aktueller Browser ([Chrome](#), [Safari](#), [Firefox](#))
8. Ggf. git-Client (bspw. [SourceTree](#)) oder Kenntnisse in git-Befehlen auf der Kommandozeile

Vorbereitungen

1. Start des Apache-Servers und MySQL-Server via Xampp (Root-Rechte ggf. erforderlich!):



2. Einspielen und Erstellen der Datenbankdatei via phpmyadmin
 - a. [phpmyadmin](#) aufrufen
 - b. Neue Datenbank per Klick auf ,neu' erstellen; ,xxs_datenbank' benennen und auf ,Anlegen' klicken:



- c. Daten per Klick auf ,Importieren' einspielen; Datei ,xxs_datenbank.sql' auswählen und per Klick auf ,OK' final importieren:



Zu importierende Datei:

Datei kann komprimiert (gzip, bzip2, zip) oder unkomprimiert sein.
Der Dateiname einer komprimierten Datei muss mit **[Format].[Komprimierung]** enden. Beispiel: **sql.zip**

Durchsuchen Sie Ihren Computer: **Datei auswählen** Keine Datei ausgewählt (Maximal: 128MiB)

Sie können auch per Drag & Drop eine Datei auf einer beliebigen Seite legen.

Zeichencodierung der Datei: **utf-8**

Teilweiser Import:

☒ Import abbrechen, wenn die maximale PHP-Skriptlaufzeit erreicht wird. (Damit ist es möglich, große Dateien zu Transaktionen zu zerlegen.)

Diese Anzahl Abfragen (für SQL) oder Zeilen (für andere Formate) überspringen, beginnend von der ersten: **0**

Andere Optionen:

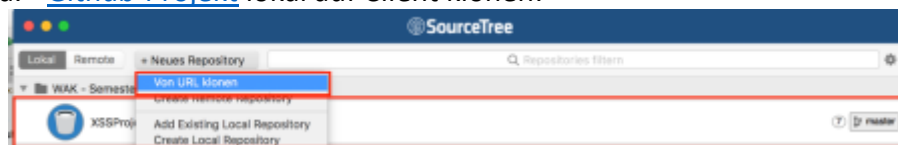
☒ Fremdschlüsselüberprüfung aktivieren

Format:

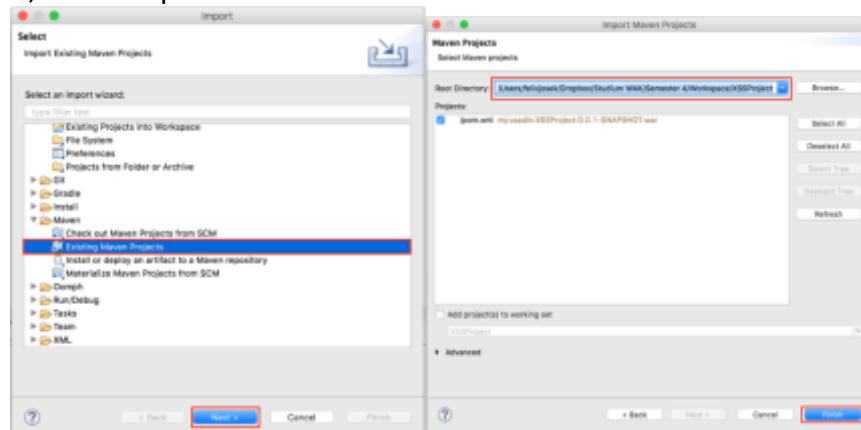
SQL

3. Import und Einspielen des Eclipse-Projekts

a. [Github-Projekt](#) lokal auf Client klonen:



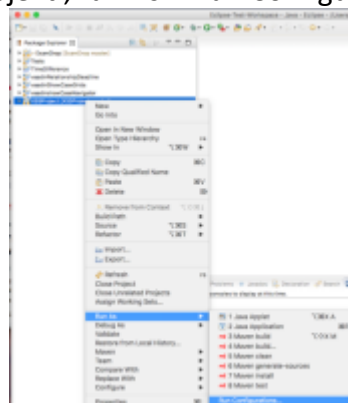
b. In Eclipse auf 'Importieren' klicken und 'Maven>Existing Maven Project' wählen. Geklonotes Projekt bei 'Root directory' wählen und per Klick auf 'Finish' importieren:



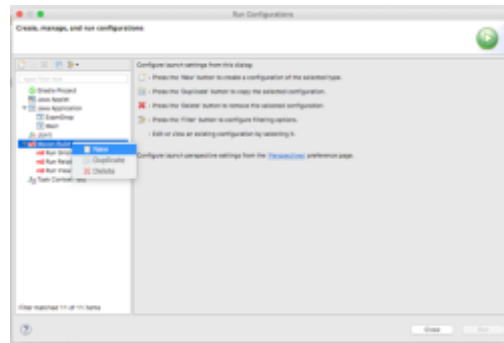
Deployment

1. Konfigurieren des Maven-Builds

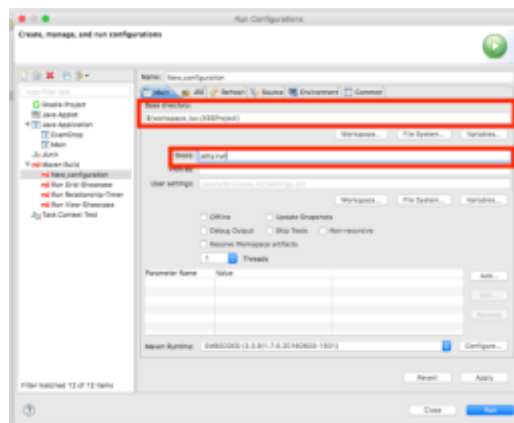
a. Rechtsklick auf das Projekt 'Run As > Run Configurations ...':



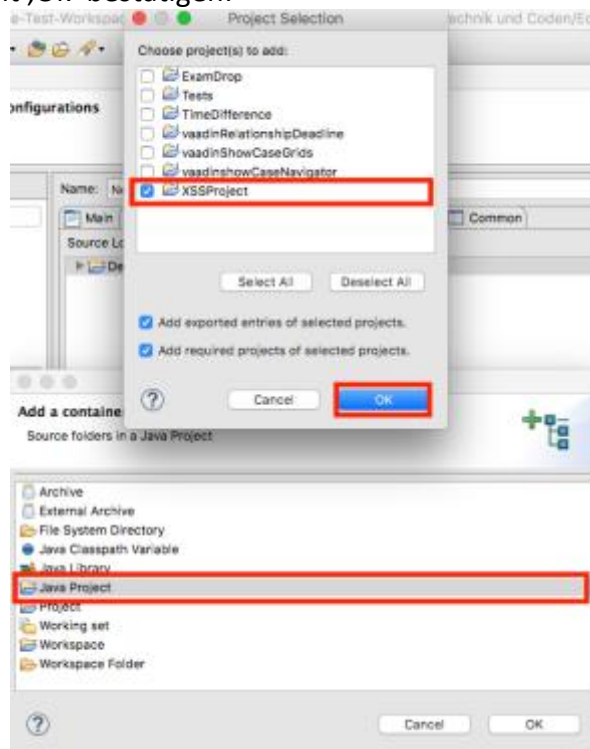
b. 'Maven Build > New...':



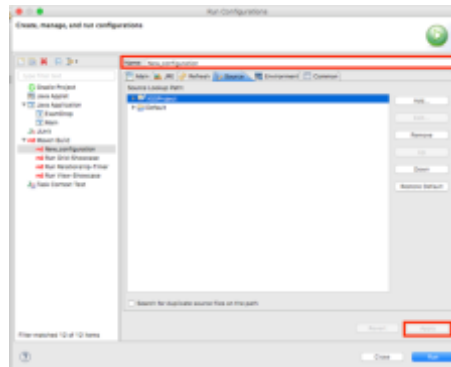
- c. Richtiges 'Base directory' wählen und bei 'Goals', 'jetty:run' eingeben. Damit wird automatisch ein kleiner jetty-Server gestartet und das Projekt darauf deployed:



- d. Auf 'Source' klicken, 'Add' drücken, 'Java Project' hinzufügen und Projekt auswählen, mit 'OK' bestätigen:

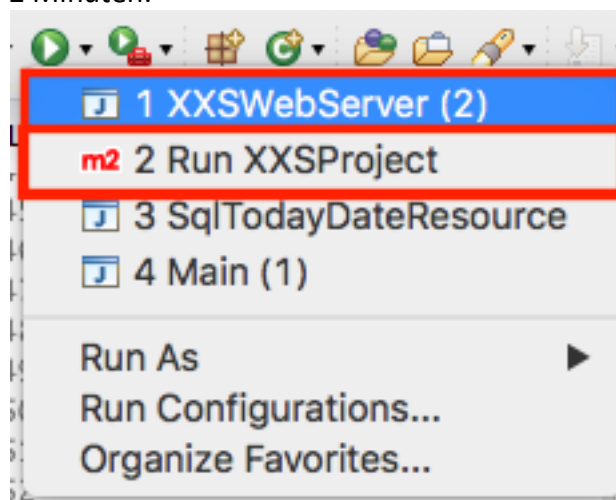


- e. Nach Bedarf Namen für Konfiguration eingeben und Eingabe mit Klick auf 'Apply' abschließen. Fenster kann geschlossen werden:

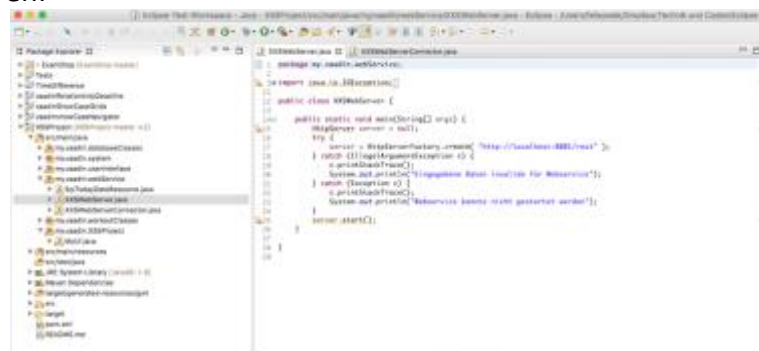


Projekt lokal bauen und ausführen

1. Webserver starten, um REST-Service bereitzustellen
 - a. Bau der Applikation mit gespeicherter Konfiguration starten. Dies dauert beim ersten Mal 1-2 Minuten:



- b. Klasse 'XXSWebserver' in 'my.vaadin.webService' öffnen und Main-Methode ausführen:



- c. Wurde die Applikation gebaut und er Webserver läuft, kann die Anwendung auf [Port 8080 lokal](#) geöffnet werden. Als Beispieluser kann 'dennsi' mit Passwort 'pumpen123' genutzt werden.