Developer Setup for web-prog. server side.

This document explains a sample Developer Setup for students attending XXXX.
You should set up your system as explained in this document, and if you get a new laptop somewhere in the middle of the course, use this document to set it up as required.

What to install and uninstall on your laptop

Un-install MySQL

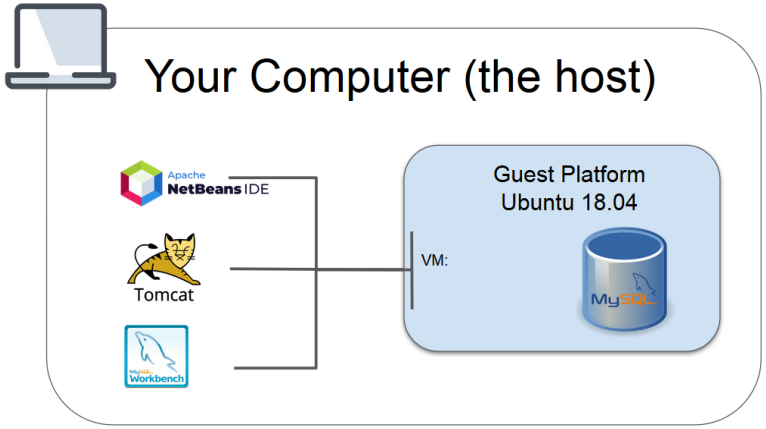
We won't insist on this part, but you will never use the local version installed before this course for anything at all this semester.
So, leave if you like, but having two versions could end up as something to create confusions.
If you uninstall MySQL, you should re-install MySQL Workbench (newest version), but NOT the server.

Install VirtualBox and Vagrant

What is happening here, and why do we do this?

We have had so many (problems with setups that did not work due to different versions of Tomcat, and MySQL, different computers (Windows, OS X, Linux) etc.

The following will ensure that you end up with a "local" MySQL server running on a Virtual Ubuntu 18.04 server (This Virtual Machine actually also has a Tomcat 9 server running, which could come in handy …).



This figure visualizes the setup on your computer when the following steps are completed.

For the rest of the course, this is the database you should use for local development. The connection parameters, user and passwords are:

```
URL:        127.0.0.1
Port:       3307
User:       dev
Password:   bx2
```


Install required dependencies to start your local Ubuntu 18.04 Server

- Download and Install VirtualBox for your OS https://www.virtualbox.org/
- Download and install Vagrant for your OS https://www.vagrantup.com/downloads.html
- Clone this repository somewhere on your system you will remember since you will access it on a daily basis for the rest of the semester: https://github.com/StartCode/vagrant-au2019a.git

Start the Virtual Ubuntu 18.04 Server

Navigate into the folder of the project you cloned in the previous step (the folder with the Vagrantfile)

In a terminal (git-bash for Windows users) execute this command: **`vagrant up`**

The first time you do this, and only the first time, it will probably take a few minutes but when completed, believe or not, you have a full virtual Ubuntu 18.04 Server, set up with the MySQL server you are supposed to use for local development for the rest of the semester.

Test your Guest OS

In the same folder, as you did vagrant up, you can do **`vagrant ssh`**

Now you are in a "remote" Ubuntu 18.04 terminal.
Try for example to execute the following command:
`mysql -u dev -p` (use the password **bx2** when requested)

You are now in a MySQL-terminal and can type SQL commands the usual way.
To see the users you get with this setup type:

```
mysql> select User from mysql.user;
```

This should print something similar to this:

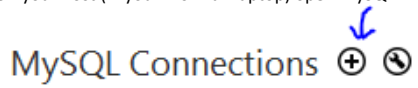
```
+------------------+
| User             |
+------------------+
| dev              |
| dev              |
| mysql.infoschema |
| mysql.session    |
| mysql.sys        |
| root             |
+------------------+
6 rows in set (0.00 sec)
```

The two *dev-users* are both granted access to all databases (the second one is allowed access from a remote server), which means you can connect to it from your own OS (From NetBeans and/or Tomcat)

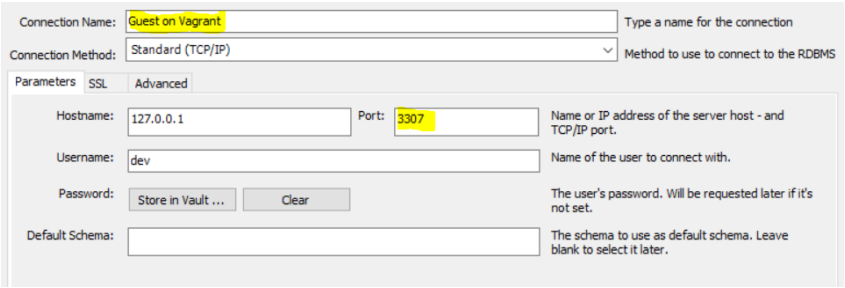
This is the user you should use **for all** local databases used for the rest of the semester.
If you prefer a terminal you can just continue like this whenever you need to create/change your "local" databases. If you prefer a GUI you can use WorkBench as explained in the next step.

Connect to MySQL server on the Guest from Workbench on the host

On your host (= your 'normal' laptop) open MySQL Workbench and create a new Connection



Add the following values in the window that pops up.
"Guest on Vagrant" for the Connection Name.
Observe that we are using the preconfigured user **dev** and the password is **bx2,** and port number is **3307**



Pres "Test Connection" and when this work press OK.

Use this connection for the rest of the semester whenever you need to create/change "local" developer databases.

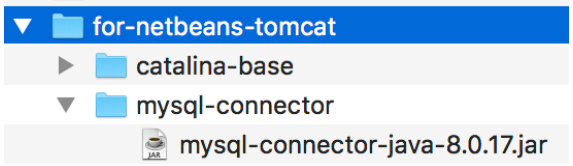
Install NetBeans and the Tomcat 9.x to be used by NetBeans

Before you install

Somewhere (read/writable) on your system create a folder called `for-netbeans-tomcat` with subfolders as illustrated below:

- **`for-netbeans-tomcat`**
  - **`catalina-base`**
  - **`mysql-connector`**

Download the MySQL Connector from here (Select ***Platform Independent*** in the "Select Operating System" dropdown), unzip and copy the file *mysql-connector-java-8.0.xx.jar* into the folder `mysql-connector`, created in the previous step.



Install Netbeans

Install NetBeans 11.1 using the installer for your OS from here:
https://netbeans.apache.org/download/nb111/nb111.html

During the install, make sure the "Check for Updates" checkbox is selected

Important when you start NetBeans If it asks whether to use settings from a previous version of Netbeans, say **NO**. We want a new clean installation

Install Tomcat 9.x

Download the zip-version of Tomcat 9 from here https://tomcat.apache.org/download-90.cgi (important DO NOT select the Windows Service installer, but the binary distribution zip-version).
Windows users:
Unpack and copy the whole folder into your Program Files folder (requires Administrator rights). Remember this location for the next step.
Mac users:
Unzip and copy the "apache-tomcat-9.0.xx" folder to the "Applications" folder.
Open the bin folder in the Tomcat application and make all the .sh files executable:
bin> chmod +x *.sh

To check if the installation is working:
start Tomcat (bin> ./startup.sh),

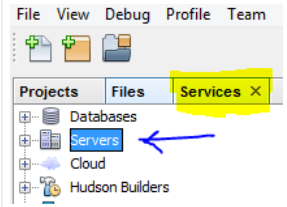
open 'http://localhost:8080' in a browser => Tomcat's index-html and

Stop Tomcat (bin> ./shutdown.sh).

Setting up NetBeans to use this version of Tomcat.

Start Netbeans 11.1 and close the welcome window

Select the *Services* tab, and right-click *Servers* and select "Add server ..."

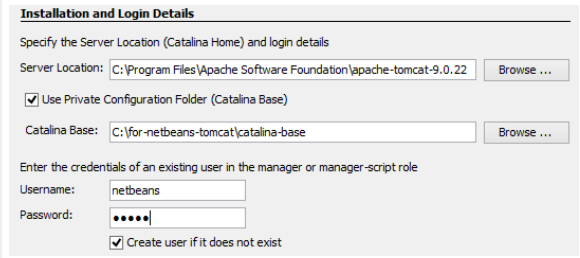


On the next window:
Select "Apache Tomcat or TomEE" and write "Tomcat 9" as the name
Press next

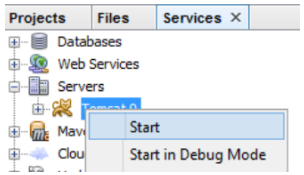
On the next window.
If this window reports "*support for Java Web and EE must be activated, select "Download and Activate...*", say Yes and accept <u>any</u> suggestions given after this
Note: On my computer, this ended in a situation where I had to stop this process. If this happens to you, just repeat the previous step, and this time it will activate the installed plugins.

On the next window, add:

The Server Location to where you un-zipped Tomcat.
Select “Use Private Configuration Folder” and add the location to the (empty) folder `catalina-base`, created in a previous step.
Add a username and password. You can write anything you like, they are only meant to be used by NetBeans, but make sure the “Create user if it does not exist” is checked.
Mac-users: Server Location is /Applications/apache-tomcat-9.0.xx



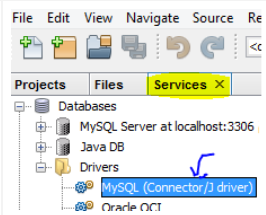
Verify that Tomcat is properly installed by starting it from within Netbeans



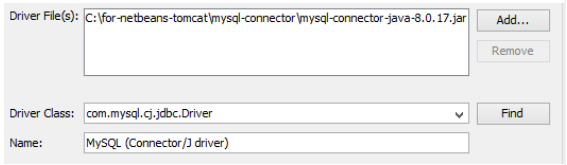
Setup NetBeans as a MySQL client

The final step in the NetBeans setup is to set it up as a MySQL client (similar to MySQL Workbench) using the MySQL server running on your Virtual Vagrant Ubuntu 18.04 Server.

In the Services tab, open Databases and the Drivers section and right-click “MySQL (Connector/J driver) and select “Customize”



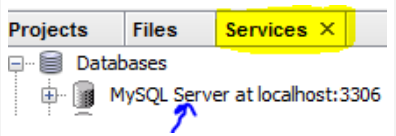
In the window that pops up:
1)
Add the full path to the driver you added in the step “Before you install”
2) (skip this part if it courses problems)
Change the name of the driver to the new name use by MySQL 8 “`com.mysql.cj.jdbc.Driver`”



Finally, setup MySQL server to point to the instance running on your Virtual Guest.
Note: This step requires you have started the Guest (vagrant up)

Right-click MySQL Server and select properties.
In the window that pops up, enter the following values:

```
Server Host Name:          127.0.0.1
Server Port Number:        3307
Administrator User Name:   dev
Administrator Password:    bx2
```



Install and setup Maven

We are going to use Maven for some of our projects this semester.

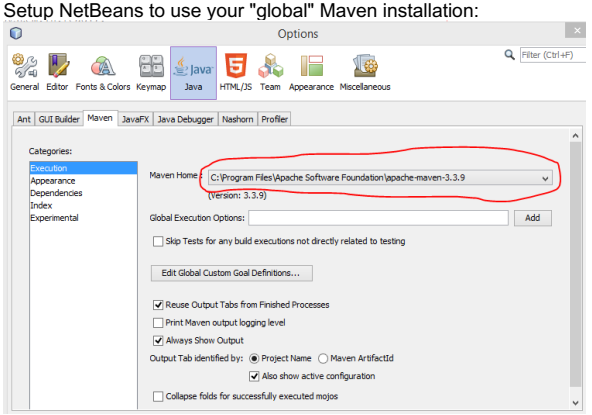
NetBeans ships with its own version of Maven, but since we are going to use Maven also from outside NetBeans, you should ensure that you have the same version, no matter where you are.

Follow the instructions below to set up your laptop for Maven Development:

Setup your "global" Maven installation

Goto https://maven.apache.org/download.cgi and download Maven.
Windows users: Use this document to install/setup Maven on your local system. Watch out for the windows tips.
Mac users: Follow the Mac section in this document
When done you should be able to run: mvn --version from command line.

Setup NetBeans to use your "global" Maven installation:

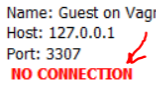


Now let's change NetBeans to use the maven installation we installed above.

Select Tools→Options, and from there select Java and the Maven tab, and change Maven Home to point to the maven installation you installed above.
Click OK, and you should be ready for Maven :-)

How to use the Setup

Most of you probably had MySQL server installed as a service which always was running. This meant that whenever you did database "things" you could just connect, either from Workbench or from your Java Code. With the setup given above, you will have to **manually** start your Virtual Server in order to use your development database. If you see this message in Workbench, or you cannot use your connections in NetBeans, its most likely because you forgot to start your Guest OS.

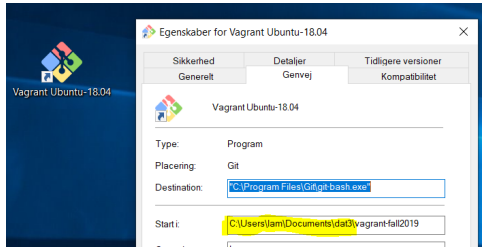


The following will provide some hints to eliminate this as a problem.

Create a quick shortcut to open a git-bash terminal in the vagrant folder

Windows

On your desktop, right-click and select **New Shortcut**. Give it a name that makes sense for example **Vagrant Ubuntu-18.04**. Provide the *Target* path to your git-bash.exe, and in *Start in* type the path to your vagrant-folder (the one you got when you cloned our code in the step "Install required dependencies to start your local Ubuntu 18.04 Server")



When you double-click this shortcut you will get a git-bash terminal in your vagrant folder.

Most common Vagrant commands	
vagrant up	This will start your Virtual Ubuntu 18.04 Server via VirtualBox. First time it will take some time, since it will have to create the Ubuntu Server, and install all our dependencies. Next time it should only take a few seconds. If you don't start your Virtual Guest like this, you won't have access to the database referred to in all exercises
vagrant halt	Use this to stop the Virtual Ubuntu Server. You will have to do **vagrant up** again, in order to use the services on the Guest.
vagrant ssh	This will ssh into your Virtual Ubuntu Server, and now you can navigate and do things exactly as you do when you navigate in a remote ssh shell.

Create a shortcut to run vagrant up, and then start NetBeans 11.1

windows

Recommendation: Start your Guest manually as explained above the first few times until you know conceptually what goes on. After that, you can simplify your daily routines by creating a shortcut that first runs vagrant up, and then starts NetBeans.

1) First create (somewhere on your system) a file **runvagrant.sh** (observe the extension and forward slashes, its meant for git-bash) with the following content (change paths to match your own system):

```
# This should CD into the folder where your vagrant file is located
cd /C/Users/???/…/…/vagrant-au2019a
vagrant up
```

2) Next, create a file on your desktop called **DB-and-NetBeans.bat** (observe the extension, this is a Windows bat-file) and add the following content (change paths to match your own system):

```
cmd /c "C:\Program Files\Git\bin\sh.exe" --login -i -- C:\Users\???\...\...\runvagrant.sh
START C:\"Program Files"\NetBeans-11.1\netbeans\bin\netbeans64.exe --console suppress
```

Since the file was placed on the desktop (skrivebord) you can just double-click it, and it will run vagrant up and then start Netbeans. If your Guest OS is already running, it will detect this, and just skip the steps that start the server