Graz University of Technology

How to Setup a Web Server for a Conversational Agent

Patrick Steyer,patrick.steyer@student.tugraz.at

Graz University of Technology
Institute of Interactive Systems and Data Science, Austria

Graz, July 2020

1 General

The current version of the Web Server can be found at `https://github.com/Tot333/WebServer_Bazaar`.

1.1 Insert your own Agent files

If you already have your own agent export the executable via Eclipse and copy it into the mturkagent folder. If you name your executable jar file other than "ReboAgent.jar", you will have to replace the text "ReboAgent.jar" in the launch script "launch_agent.sh" with the name of your jar file. Also for your agent to work as expected, copy all the files and folders of the runtime folder also into the mturkagent folder. Make sure no files except the launch script and the runtime files of your project remain in mturkagent.

Since your agent will have its own name make sure to set the name in the "Agent.xml" and in the "WebsocketChatClient.properties" file correctly. This will be important so that the agent joins under the desired name and to not get into a possible infinity loop of agent instances.

Also in the "WebsocketChatClient.properties" file replace the URL to the URL of your server instance. Side note: It seems there is a difference between the TUGraz network internal and external URL. It seems that using the internal URL is the right choice!

2 Install Docker

The first step to set up your own web server for a conversational agent is to install the packages docker and docker-compose. The three important components for your web server will run in Docker images. Those three components are:

- the server files
- the database
- the proxy

To install docker use the newest version from their website `https://docs.docker.com/install/linux/docker-ce/ubuntu/`. Follow the instructions given and when you are done installing make sure to test docker with the help of the typical Hello World example. A tutorial which guides you through this process can be found at `https://docker-curriculum.com/`.

Next you will need to install the docker-compose package. For this just use the link `https://docs.docker.com/compose/install/` and follow the instructions given on the website.

3 Component Explanation

In this section I'll quickly introduce the three images of the docker container which should run at the end of this setup tutorial.

3.1 Server

For the first image we have the main server which runs with node.js and handles all the traffic/messages between the server and the client/browser. For this the image needs all the files of the "bazaar" folder which get automatically copied into the image through docker.

When the image starts, it starts the runscript from that folder as a process. It is important to know that this script should run infinitely. If you have a look inside this script you will see that it mainly runs the JavaScript file "server_bdemooc_xu.js".

In this file replace all occurrences of "Rebo", inclusive the quotation marks, with the name of your agent. This prevents the server from getting in a loop of starting up new instances of your agent, since it also joins the chatroom. (Should not be a problem anymore since only new instance is generated when the first join in a chatroom happens.)

3.2 Database

For the database a mysql database is used. The structure can be found in the bazaar folder in the file "nodechat.sql".

If you set up an automatic backup with crontab with the command given in Section 5 make sure to edit the "backup.sh" so that the path to the right folder is given. Also if you want to use the backup.sh manually make sure to adapt the file according to your system.

3.3 Proxy

Next we have the proxy which makes sure that all connections are routed correctly to the web server. It routes the accesses to the website to the right ports for the the other docker images. If wanted you can get rid of this docker image by setting up the proxy on the server on your own.

4 Start the Web Server

Make sure that before you execute the commands that you are in the right folder, which is the outermost level of the repository. There are for instance the folders bazaar, ha-proxy and mturkagent located.

Before starting up the docker images we have to rid of the problem with access rights in the files for the docker images. For this you just have to execute the command

```
sudo chmod -R a+rwx <folder>
```

This command recursively changes all files so there are full access rights on them.
Now for starting up the server you will need execute the following commands:

```
docker-compose -f docker-compose-dev.yml build
docker-compose -f docker-compose-dev.yml up -d
```

This could take some time since a lot of files are downloaded to create those three images. If you want to have an output or log of the docker up command just strip the -d of the command and you will see all logs. Also you are stuck in the terminal then until you press STRG+C, which also ends the images again. After you have run the two commands successfully, execute

```
docker ps
```

which lists all the running images and should list three images which are running. If not all of them are listed add the option -a to the previous command, which lists also terminated images with their exit codes.

Since it always can happen that the server crashes or something unexpected stops the docker container, I recommend to set the images to restart always if ended unexpectedly. To do this run the following command for all three images. You can see the name of the images with the command used before or you can just use the typical Linux auto complete.

```
docker update –restart=always <name>
```

5 Useful Commands

In this section i will list some useful commands with an explanation when to use them.

- docker system prune; docker rmi $(docker images -a -q)
  If you have to clean up all the images and containers run the two commands which should clear the complete image and container storage. Can also be useful to get rid of all the intermediate images docker-compose creates, only take the second command for this.

- docker exec -it <name> /bin/bash
  This commands will connect you to the terminal of the selected docker image. Can be quite useful to check if certain scripts are executed or you want to have a look at the database. To get out of the terminal just write exit and you will be back in the terminal of your server instance.

- mysql -u root -p
  When you are in the docker image for the database execute this command to get access to the database. Then you can for instance clear the tables or make a backup.

- sudo chmod -R a+rwx <folder>
  Since the JavaScript script of the server starts up instances of your agent you will have to change the mode of those files a little bit or else the correct access rights are not given. Since I didn't track down exactly which files all need this setting to work i just gave it to all files in the repository.

- sudo crontab -e
  Crontab is a tool which helps you to execute scripts or commands at certain times. I've used it to let the server restart each Sunday to prevent the server with pilling up stuck agent processes. Also I've made a task to automatically backup the database each week.

6 Website different URLs

The next feature that was implemented is the possibility to access the website through three different modes. The first mode is for the chat with Rebo, the second for writing a small essay and the last for a group discussion. For the chatbot and the group mode the website looks and behaves the same, with the only exception that in the group mode the discussion is not led by a chatbot and multiple users have the possibility to chat with one another. In the essay mode the website is slightly modified, so that an essay can easily be written. The mode parameter in the URL was created as a replacement for unused parameters. The URL to access the chatrooms is now constructed as follows: `http://rebo.know-center.tugraz.at/bazaar/chat/<chatroom>/<chatbot|essay|group>/<name>/`, the <chatroom> placeholder gets automatically replaced with a unique string for each conversation and the placeholder <chatbot | essay| group> shows the three modes which can be used. F

6.1 Further Useful URLs

Here we will have a quick look on a few different useful URLs of the webserver and what their functionalities are.

- http://rebo.know-center.tugraz.at/bazaar/data/<ChatRoomName>
  By using this type of URL you can export the chat data of a single chat room in CSV Format.

- `http://rebo.know-center.tugraz.at/bazaar/room_status_all`
  If you don't know the name of a chatroom you can just call this URL which lists all chat rooms in the database. Simply copy paste a name of your choice from the list into the last URL and you will get the csv file.

- `http://rebo.know-center.tugraz.at/bazaar/AllData`
  Next we have the probably most interesting command, which exports basically the whole database in csv format. The data is sorted by the chatroom name and then by the time so that all the messages which should be together are also listed one after another and not mixed through. For me it was the case that excel had problems while importing umlauts, therefore I used the makro suggested at this website `http://www.excel-ist-sexy.de/unicode-umwandeln/`. Or as alternative open the csv and save it with UTF-8 encoding.