BlueChat is a web-based chat application that will support a large number of client sessions simultaneously. It is built on top of the highly performant Grails platform and boasts a very responsive front-end built around javascript and jQuery so that it performs like a desktop application with nearly no full-page refreshes.

The BlueChat application can be executed in two different ways depending on your intention. Firstly, the application can be run in a virtual machine (VM) container completely self-contained and configured (as you may want to do in a production environment) and secondly as a development/testing environment so that compile/build/test cycles can be executed independently (as well as a run cycle for interacting with the application in the same way you'd interact with the VM version).

The procedures to stand up both versions are quite different but will be detailed in this document step-by-step. The BlueChat application has been tested on both Mac OS X 10.10 (Yosemite) and Ubuntu 14.04. It is highly unlikely that it will work in a Microsoft Windows environment without further development.

To start we'll configure and run BlueChat in a VM. In this scenario there are three prerequisite applications that need to be installed: Oracle Git, VirtualBox and Docker.

Git is an opensource Source Code Management (SCM) System for storing programming code and controlling versions of it.

VirtualBox is a piece of software that enables you to run other operating systems (OS) inside of your existing operating system.

Docker is an opensource application that downloads pre-configured OS images and allows you to make configuration changes to that OS image (such as installing additional software) in order to create a new image configured for your specific needs. That image can then be shared with others so that deploying a new application becomes a simple matter of sharing your new OS image so that no more configuration is needed, avoiding the long, repeated process of installing all the dependencies an application may need (such as a database, web server, application server, dependent libraries etc.).

Running BlueChat in Docker (running as a self-contained VM)

First we'll install VirtualBox. Since the procedure to install VirtualBox is dependent on your host operating system, it's best to visit https://www.virtualbox.org/wiki/Downloads and follow the instructions for your host machine's operation system.

To install Docker, visit https://docs.docker.com/installation and again follow the instructions for your host machines operating system.

Finally we'll install Git. Following the instructions at http://git-scm.com/book/en/v2/Getting-Started-Installing-Git for your particular operating system.

Some of the steps needed to run BlueChat vary depending on if you're running on Mac OS X or Ubuntu Linux 14.04. Where there is a difference it will be brought to your attention.

Now it's time to install the bootstrap file for standing up the Docker instance of BlueChat. The steps are as follows:

1) First you need a directory in which to work. Wherever you see fit, execute the following command to create a new directory: *mkdir mybluechat*
2) *cd mybluechat*
3) Now we'll get the Docker bootstrap configuration file from a git repository. In the "mybluechat" directory execute: *git clone https://github.com/kathsoftware/BlueChatDocker.git*
4) After step 3 is finished, run *cd BlueChatDocker*

In this directory is a file named "Dockerfile", do not change the file in any way. It's from this small script that the entire Linux environment and BlueChat application will be configured.

First we'll see how to stand up the new environment on Mac OS X. If you're an Ubuntu user feel free to skip ahead to the "**Running Docker On Ubuntu**" section.

Running Docker on Mac OS X

Mac OSX needs a bootstrap VM in order to work with Docker. That bootstrap is called "boot2docker". There are a few extra steps than are needed on Linux, but it's still an easy process.

In the directory, mybluechat (the same directory where we find the "Dockerfile" file), enter the following commands:

1) *boot2docker init* (this will create the bootstrap VM. If you've already run this command, you don't ever need to run it again.)

2) *$(boot2docker shellinit)*

Type the command in exactly as shown. This will set your Docker environment variables (Note: this is only needed on Mac OS X. The output from this command should be jotted down; we'll be using the value of the environment variable DOCKER_HOST in order to access BlueChat from a browser. If you forget to jot it down right now, you can always find it out later by executing "echo $DOCKER_HOST")

3) *boot2docker start*

This command will start our bootstrap VM. Let the command run until you're returned to the command prompt. (While waiting you'll see this output: "Waiting for VM and Docker daemon to start... .....ooooo Started.")

They are the extra steps needed for Mac OS X. From this point on the commands are the same for Mac

OS X as they are for Linux. Continue on to the next section **Running Docker On Ubuntu.**

Running Docker On Ubuntu (Or if you're continuing from the Mac OS X instructions)

Note that during the next few steps you're going to see a lot of software being installed. Do not fear; the software is not being installed on your host machine but in an OS image that Docker will download onto your machine to automatically be run in VirtualBox.

4) *docker build -t bluechat . (Don't forget the dot "." character at the end!)*

This process will take about 15 minutes depending on the speed of your machine and network connection. While you wait for it to finish you'll see a bunch of "Downloading" and "Download Complete" messages. Just wait until it's finished and the command prompt comes back.)

Example:

```
Sending build context to Docker daemon 48.64 kB
Sending build context to Docker daemon
Step 0 : FROM jamesdbloom/docker-java7-maven
Pulling repository jamesdbloom/docker-java7-maven
febf1459d9ec: Pulling dependent layers
511136ea3c5a: Download complete
b3553b91f79f: Download complete
ca63a3899a99: Download complete
ff01d67c9471: Download complete
....
(When all done is will display something along the following:)
Removing intermediate container 85d570c3b4f4
Successfully built ae2073019c03
```

Your local image should now be built (but not yet running) and ready for execution. Once we start it, on Linux it will run directly under VirtualBox control but on MacOSX it will run as a VM inside of the boot2docker-vm VM that you created earlier. Either way our commands will be the same

5) Now comes the fun part. In this step we'll start up our VM image and then be able to access the BlueChat application. Enter the following command:

docker run -i -p 8080:8080 bluechat /opt/bluechat/grailsw run-app

This process will take about 7 minutes to complete depending on your machine speed and network connection. Wait for this message to appear:

```
... finished configuring Spring Security Core
| Server running. Browse to http://localhost:8080/BlueChat
```

The VM is now running and the BlueChat Grails application is running inside of it. Later when we're finished running the BlueChat application, simply ctrl-c in this console window and the VM will shutdown.

6) Okay, we're there now! Time to try out BlueChat. If you're on a Mac, you'll need to use the IP Address that you were shown when you executed $(boot2docker shellinit). If you forgot what the IP address, simply execute “echo $DOCKER_HOST” to get the IP Address. If you're on Linux you can simply use “localhost” from your while on your host machine. (Note: much further configuration would be needed in order to map the IP Address so that a web server, such as Apache, can access the BlueChat website inside the VM.)

Open a browser and go to: http://localhost:8080/BlueChat (on Mac substitute “localhost” with the IP Address as described earlier). With good luck (and the power of Docker) you should now be on the BlueChat Login page.

You can register as a user or you can sign in as one the following previously configured users:

remy/max
max/remy
sjang1/jang

Now that we've tried out BlueChat, control-c in order to stop the VM that Docker is running for us. Feel free to go back and try this process again if you choose. Now we're going to do things a bit more directly so that we can compile, deploy and run unit tests against BlueChat.

Running BlueChat Directly On Our Machine

This process will allow us to run BlueChat in the traditional way; directly on our machine. It simply involves getting the sourcecode from the Git repository and executing some Ant scripts against it.

Prerequisites.

In order to run BlueChat directly on our machine we need three additional pieces of software: Oracle Java JDK7, Grails and Ant. Note that while the previous process didn't install any additional software on our machine (other than VirtualBox and Docker), this process will require the installation of additional software directly on our machine.

To install the Oracle JDK7, go to http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html and follow the instructions for your operating system. Be sure to set the environment variable “JAVA_HOME” that points to your new JDK7 installation and add $JAVA_HOME/bin to your PATH environment variable.

To install Ant:

On Linux: sudo apt-get install ant
On Mac OS X: brew install ant (if you have homebrew installed, which you should)

To install Grails:

Download Grails 2.4.4 at http://grails.org/download
Be sure to set the environment variable “GRAILS_HOME” that points to your new JDK7 installation and add $GRAILS_HOME/bin to your PATH environment variable.

Now that you have JAVA 7, Grails, Ant and Git installed (Git should still be installed from the "Docker" part of this tutorial), we can move on to getting the sourcecode.

1) Create a directory anywhere on your machine: *mkdir testbluechat*
2) *cd testbluechat*
3) *git clone https://github.com/kathsoftware/bluechat.git*
4) Now we'll run the BlueChat application. At the command prompt enter: *ant run*

The application will now start. Once you see the message "Server running. Browse to http://localhost:8080/BlueChat", you can use your browser to access the application at http://localhost:8080/BlueChat. It should behave just as it did when you tried it in the Docker exercise.

5) When you are done you can control-c to stop the application.
6) To clean, compile and unit test the application, execute: *ant test*

Once all of the tests have run, you should see a message stating that the tests have all "Passed". Logging messages can be seen in the "applogging.log" file.

The directory in which you are sitting is also an Eclipse project file. The project can be opened with the Eclipse IDE or any standard text editor in order to examine the code.

That concludes our tutorial with BlueChat. I hope that all has gone smoothly for you and everything worked as it should.

Thank You!