

|  |  |
| --- | --- |
|  | Abstract  This document describes the step by step installation of Eclipse Codewind plugin and Docker Compose Container development, including test examples, using Eclipse Version: 2019-06 (4.12.0), installed on the Linux Master node on the IBM ECM Cloud Private Container on RHEL 8.0 .   1. Alan Bluck |

© ASB Software Development Limited

Contents

[IBM Installations - ASB Software Development Limited Publications 3](#_Toc20139075)

[Installation of Eclipse Codewind and an Example using Docker Compose on RHEL 8 Linux 4](#_Toc20139076)

[Create an example Docker Container 15](#_Toc20139077)

[Create a Dockerfile 17](#_Toc20139078)

[Define services in a Compose file 18](#_Toc20139079)

[Build and run your app with Compose 18](#_Toc20139080)

[Edit the Compose file to add a bind mount 24](#_Toc20139081)

[Experiment with some other commands 27](#_Toc20139082)

[Appendix A – Example Test Docker Container 28](#_Toc20139083)

[Appendix B - Example Eclipse plugin python container log 31](#_Toc20139084)

## IBM Installations - ASB Software Development Limited Publications

<https://doi.org/10.13140/RG.2.2.20160.69129>

IBM FileNet P8 Java Development on ECM Cloud Private Container P8 Examples

(**NB Above describes the installation of the Eclipse IDE I used for this document**)

<https://doi.org/10.13140/RG.2.2.27358.18246>

IBM Cloud Private P8 Container CPIT Installation on RedHat Enterprise Linux 8.0

(**NB Above describes the installation of the base system I used for this document**)

<https://doi.org/10.13140/RG.2.2.22030.92486>

**Problem Resolution Procedures For fixing Software Installation Issues**

<https://doi.org/10.13140/RG.2.2.27345.89440>

**IBM BAW 18.0 Installation phase1 preprint with install of IBM Workflow Center 8.6.1.19002**

<https://doi.org/10.13140/RG.2.2.10491.67369>   
**DB2 10.5 Installation on RHEL 8.0 V1**

<https://doi.org/10.13140/RG.2.2.33527.57761>

**IBM Security Directory Services 6.4- Installation on RHEL 8.0**

<https://doi.org/10.13140/RG.2.2.15007.10408>

**WebSphere 8.5.5.15 Installation 29-06-2019 on VMWare Workstation Pro 15.1**

<https://doi.org/10.13140/RG.2.2.15737.83048>

**IBM BAW 18.0 Installation 18 07 2019 - Install of IBM Security Directory Suite 8.0.1.1 on RHEL 8 using VMWare Workstation Pro 15.1**

<https://doi.org/10.13140/RG.2.2.21708.16001>

**Case Manager 5.3.3 Installation on RHEL 8.0 with Content Navigator 3.0.6**

<https://doi.org/10.13140/RG.2.2.31489.10082>

**Installation of Oracle 12C on the Linux operating system**

<https://doi.org/10.13140/RG.2.2.14590.95049>

**Content Navigator 3.0.6 Installation on RHEL 8.0\_V1.docx**

<https://doi.org/10.13140/RG.2.2.21170.76480>

**Content Foundation 5.5.3 Installation on RHEL 8.0\_V2.docx**

<https://doi.org/10.13140/RG.2.2.30401.51048>

**Case Manager Installation on RHEL 8.0\_Preparation.docx**

This document provides step by step Installation steps to install

IBM FileNet Content Engine 5.5.2

# Installation of Eclipse Codewind and an Example using Docker Compose on RHEL 8 Linux

REF:

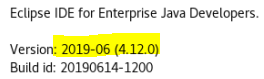
<https://www.eclipse.org/codewind/mdteclipsegettingstarted.html>

<https://docs.docker.com/compose/gettingstarted/>

Download and install the latest [Eclipse IDE for Java EE Developers](https://www.eclipse.org/downloads/packages/release/) or use an existing installation.

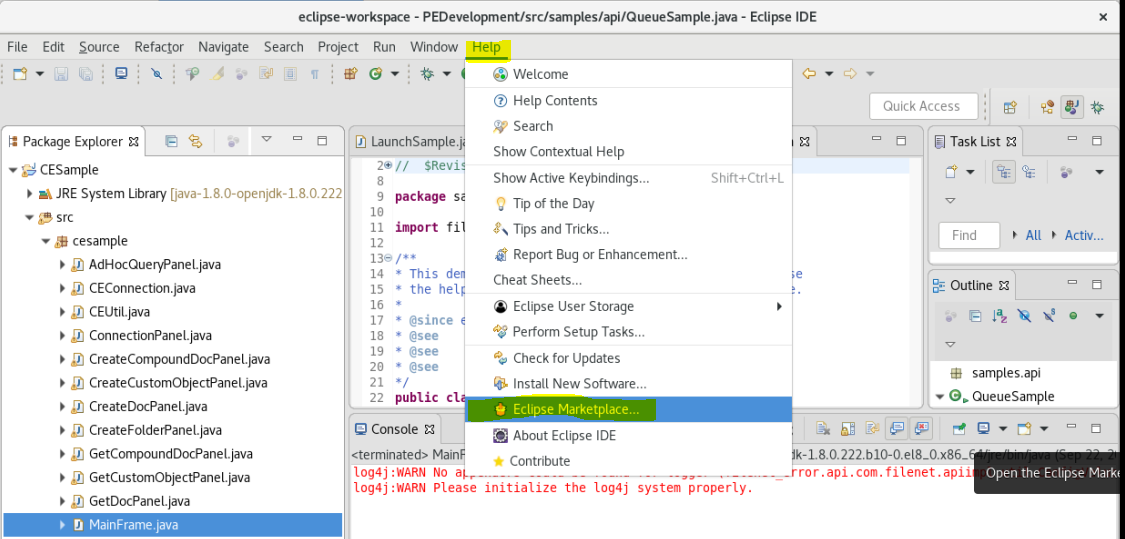
* Install Eclipse IDE Version 2019-09 R (4.13.0) or later to avoid [Bug 541220](https://bugs.eclipse.org/bugs/show_bug.cgi?id=541220).

**I used the current installed:**

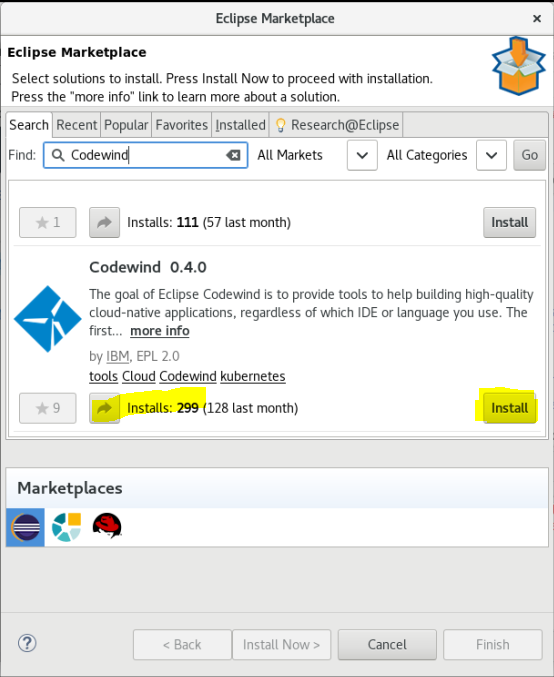


However, the earliest supported version of the Eclipse IDE is Version 2019-03 (4.11).

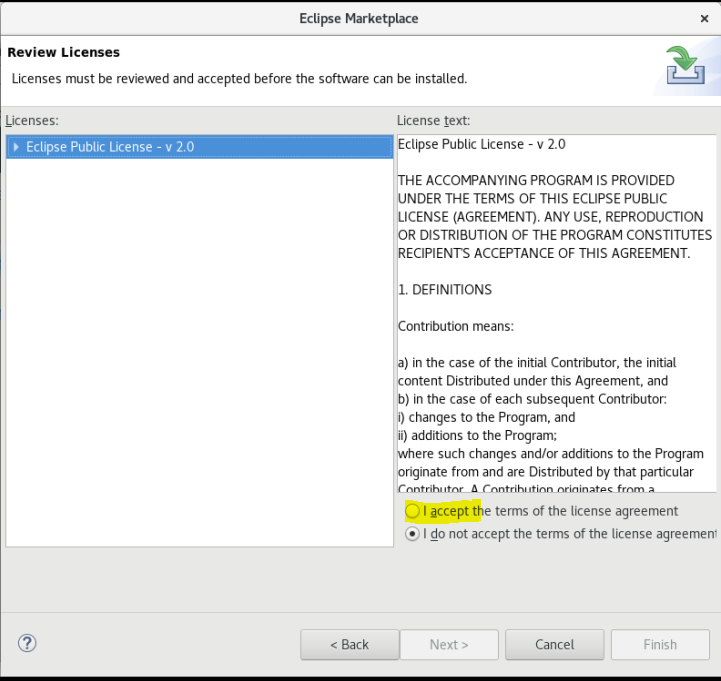
1. Open the Eclipse IDE and navigate to **Help** > **Eclipse Marketplace**.



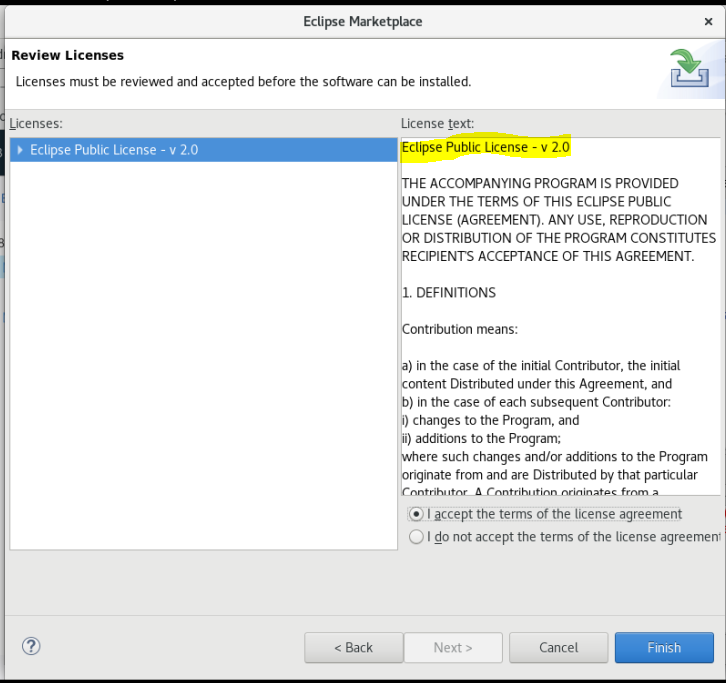
1. Search for **Codewind**.



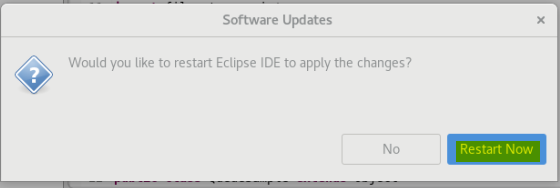
1. Click the **Install** button.



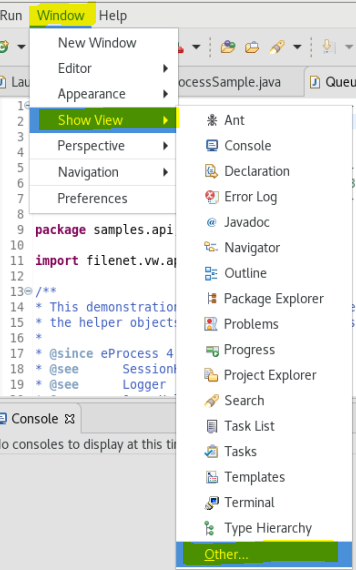
Click Accept



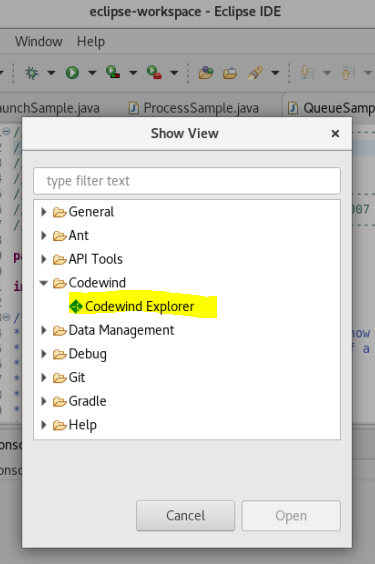
1. Finish the wizard and accept licenses as needed.

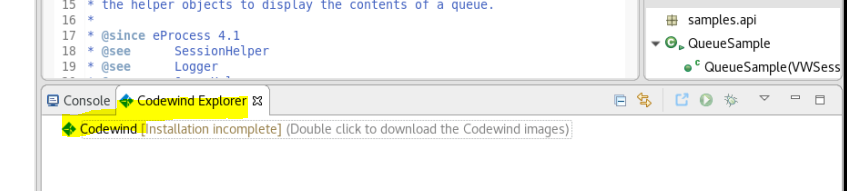


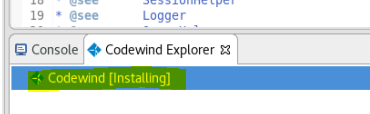
1. When the installation is complete, restart Eclipse.
2. Open the Codewind view. Navigate to **Window** > **Show View** > **Other…** > **Codewind** > **Codewind Explorer**

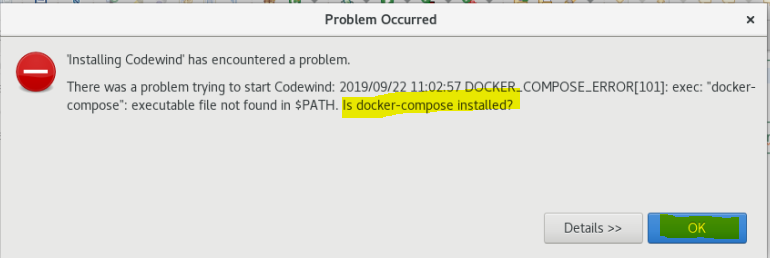


1. Codewind requires the installation of additional Docker images to run. Double-click on the **Codewind** item in the Codewind Explorer view to complete the installation. The installation may take a few minutes to complete.







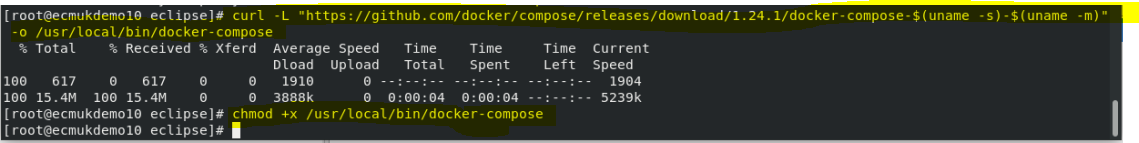


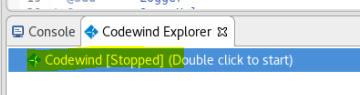
There was a problem trying to start Codewind: 2019/09/22 11:02:57 DOCKER\_COMPOSE\_ERROR[101]: exec: "docker-compose": executable file not found in $PATH. Is docker-compose installed?

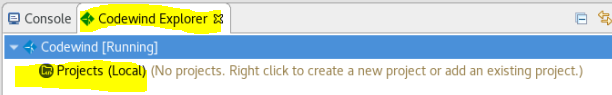
<https://docs.docker.com/compose/install/>

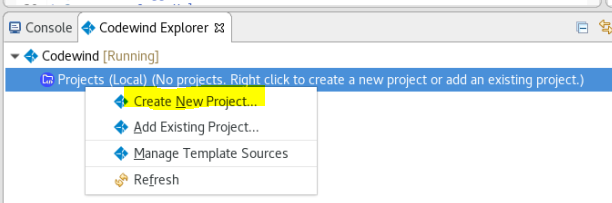
curl -L "https://github.com/docker/compose/releases/download/1.24.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

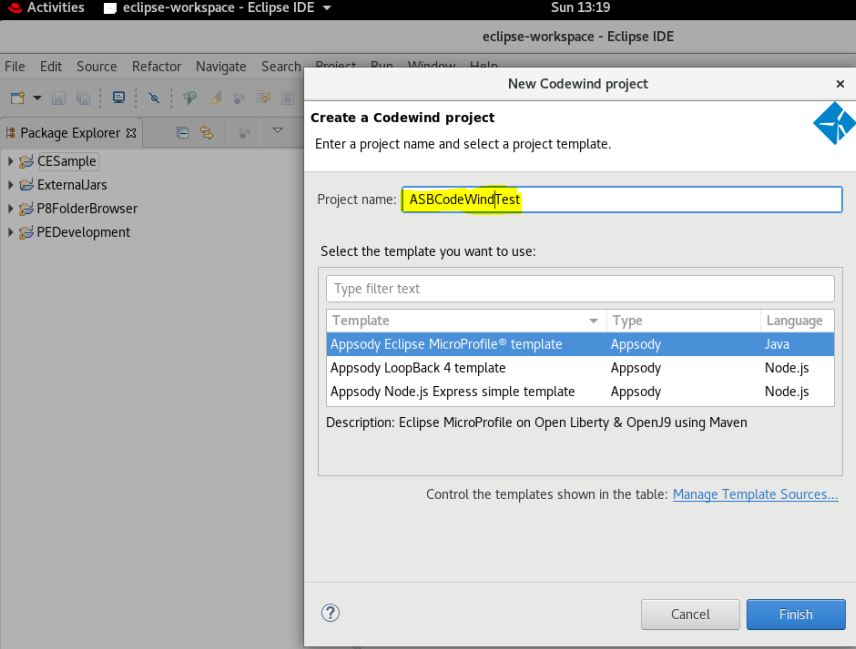
chmod +x /usr/local/bin/docker-compose



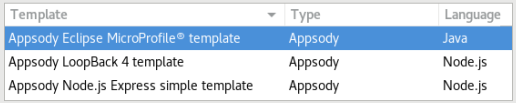


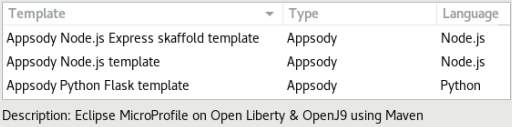


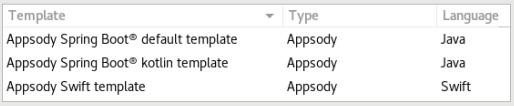


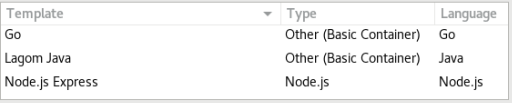


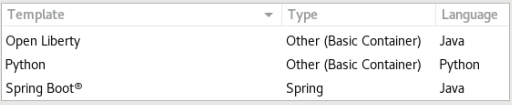
Select the project type required from the scrollable list:

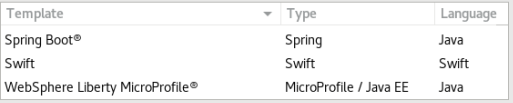


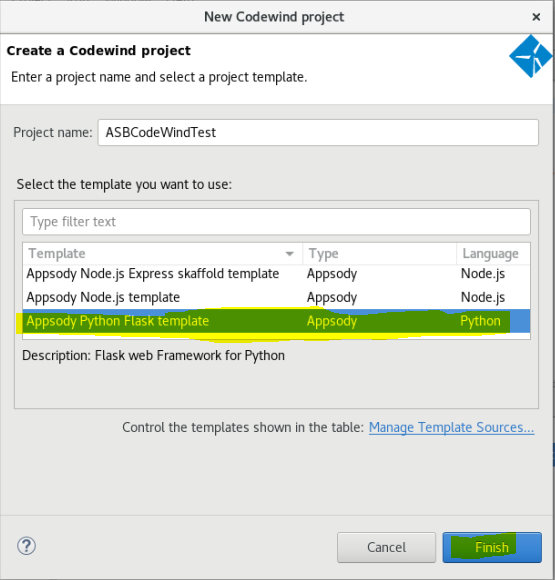


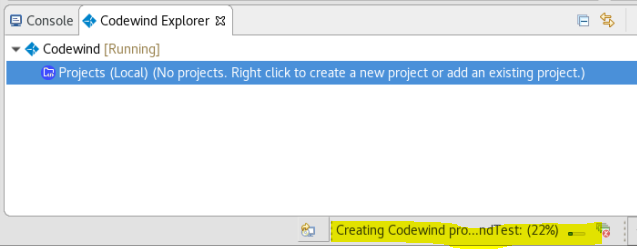




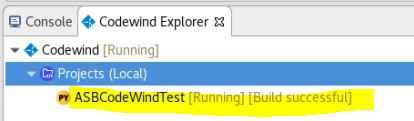




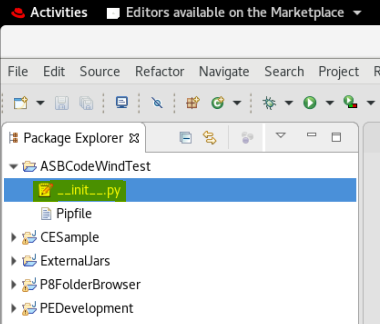


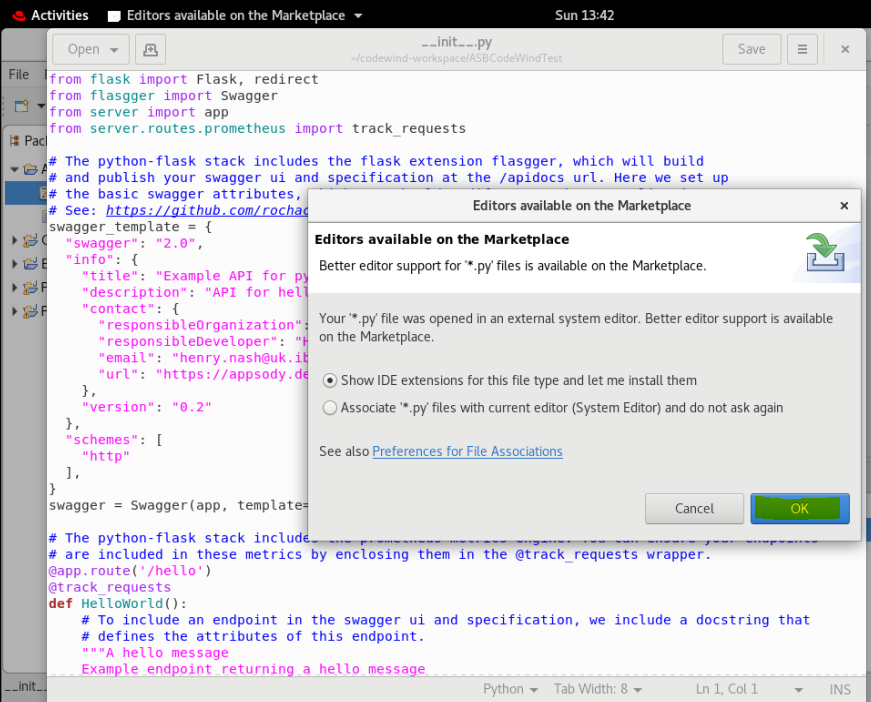


Wait for the template build for the Project name we created

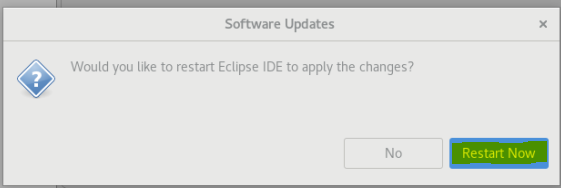


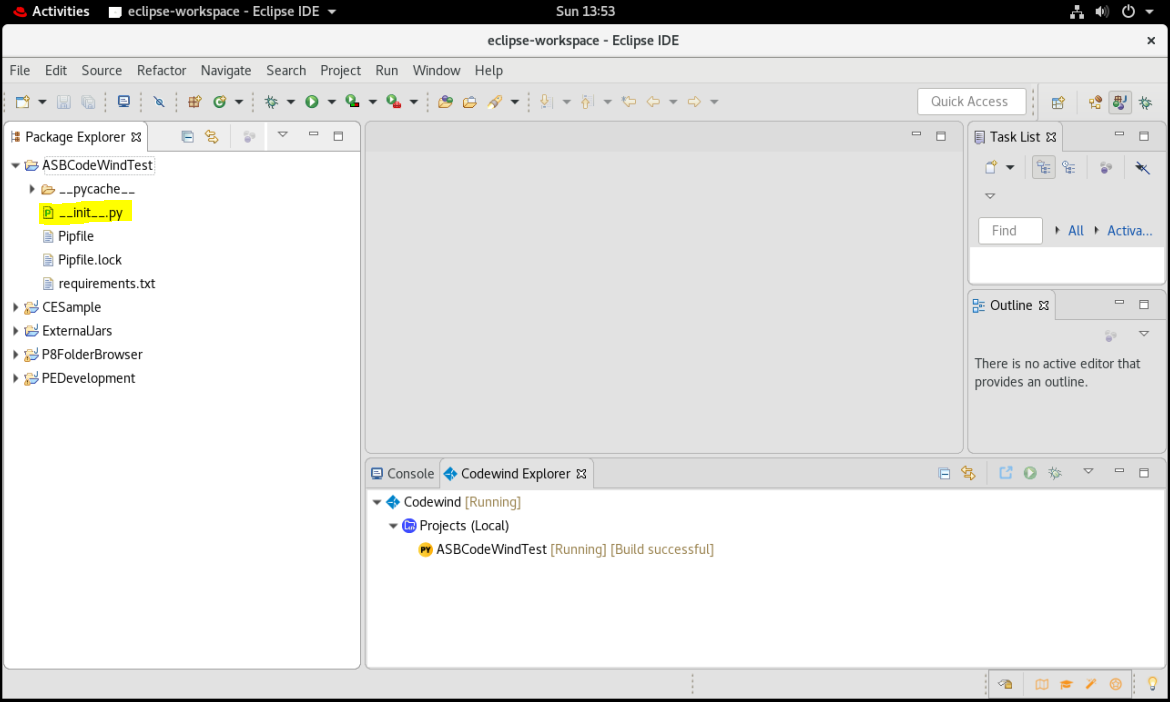
Double-click on \_\_init\_\_.py

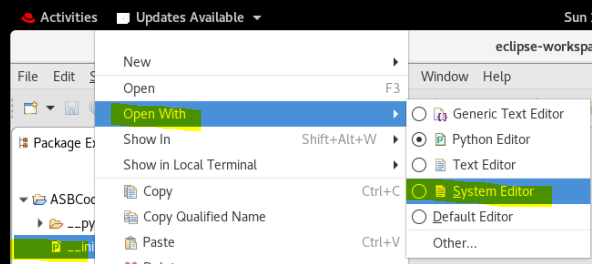




I suggest Cancel as the plugin-editor selected gives a Java Null-pointer exception!

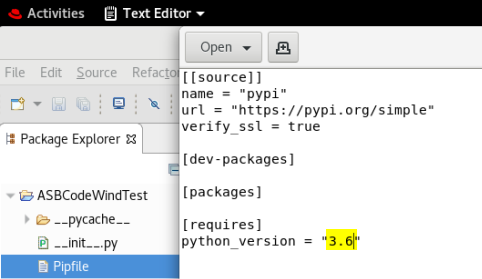




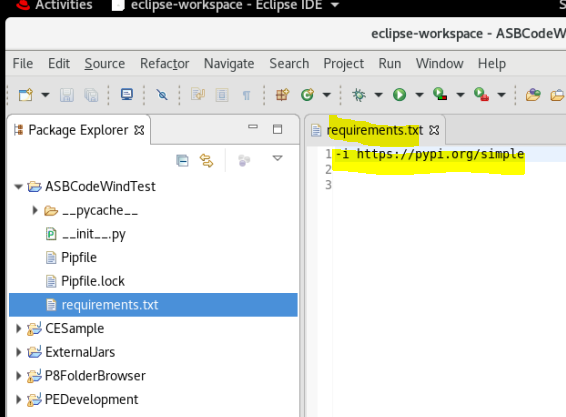


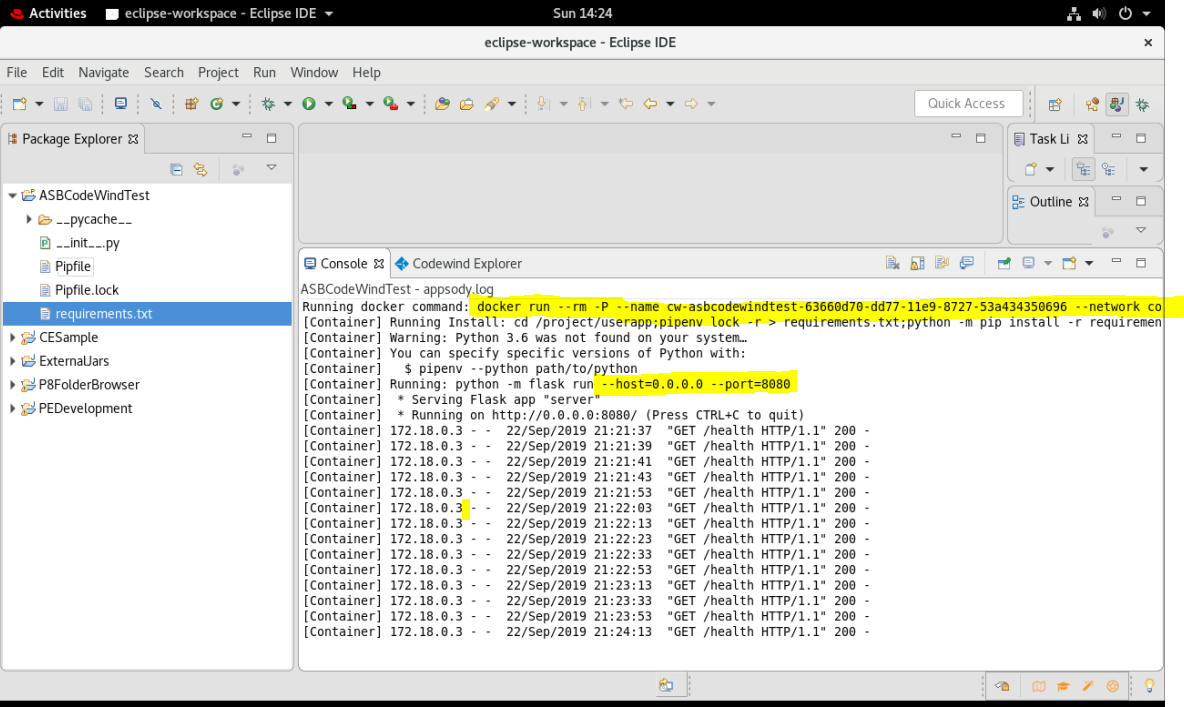
**Above Python Editor defaulted, has an issue!**

**Selected System Editor**

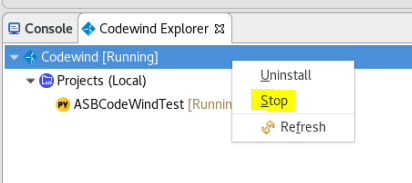


**Changed from 3.7 to 3.6**

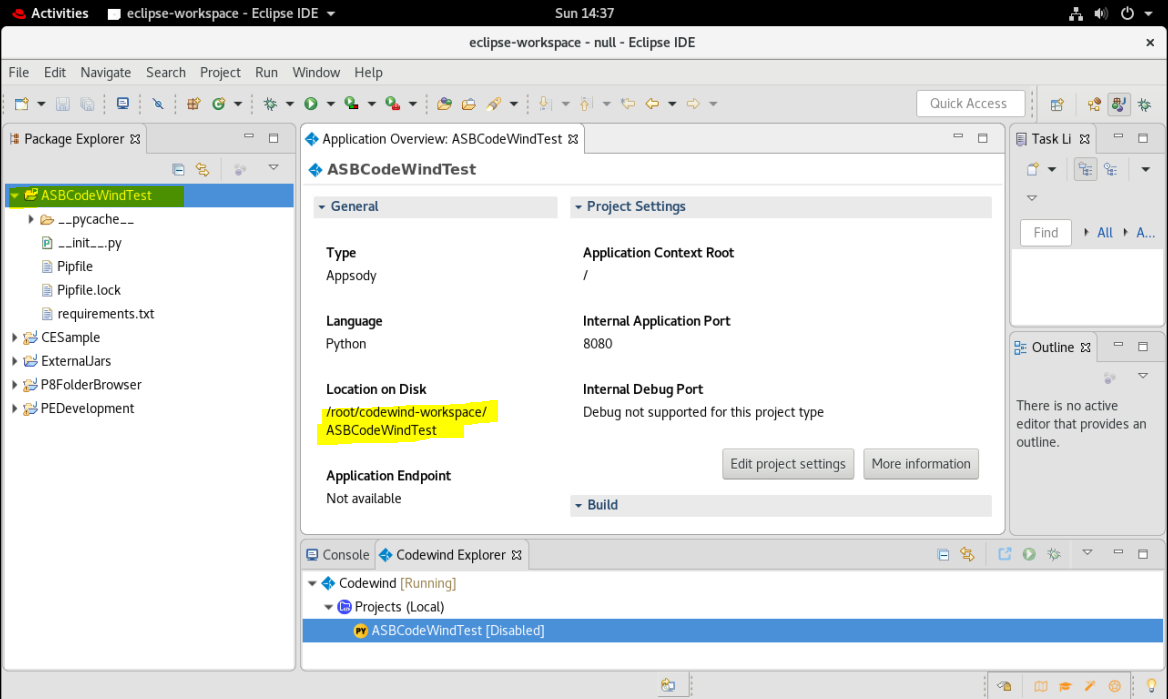




See Appendix B Example Eclipse plugin python container log project log for full list



Right-mouse click on the Codewind [Running] node and select Stop



Disable project to stop this running and right click to view the Application details

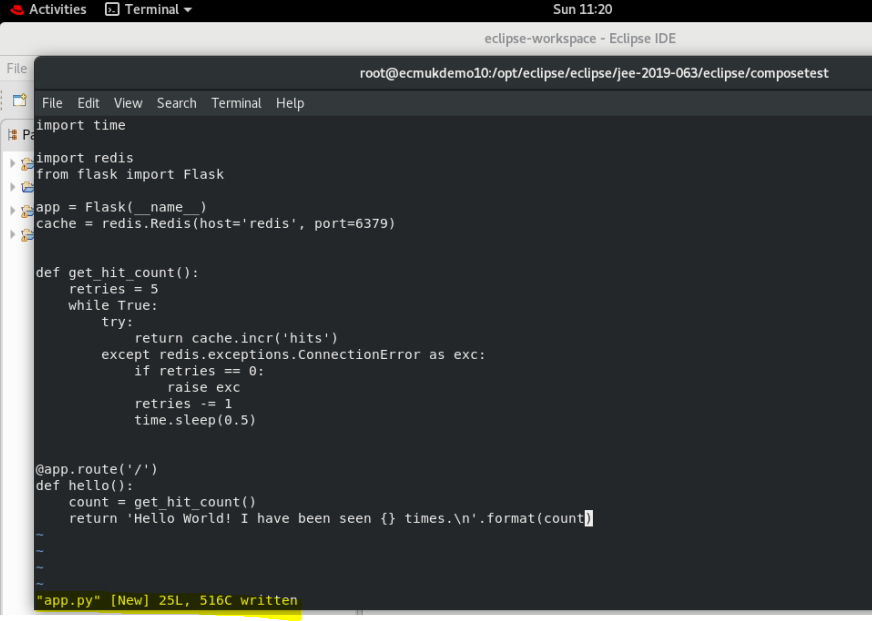
# Create an example Docker Container

REF <https://docs.docker.com/compose/gettingstarted/>

# mkdir composetest

# cd composetest

vi app.py



import time

import redis

from flask import Flask

app = Flask(\_\_name\_\_)

cache = redis.Redis(host='redis', port=6379)

def get\_hit\_count():

retries = 5

while True:

try:

return cache.incr('hits')

except redis.exceptions.ConnectionError as exc:

if retries == 0:

raise exc

retries -= 1

time.sleep(0.5)

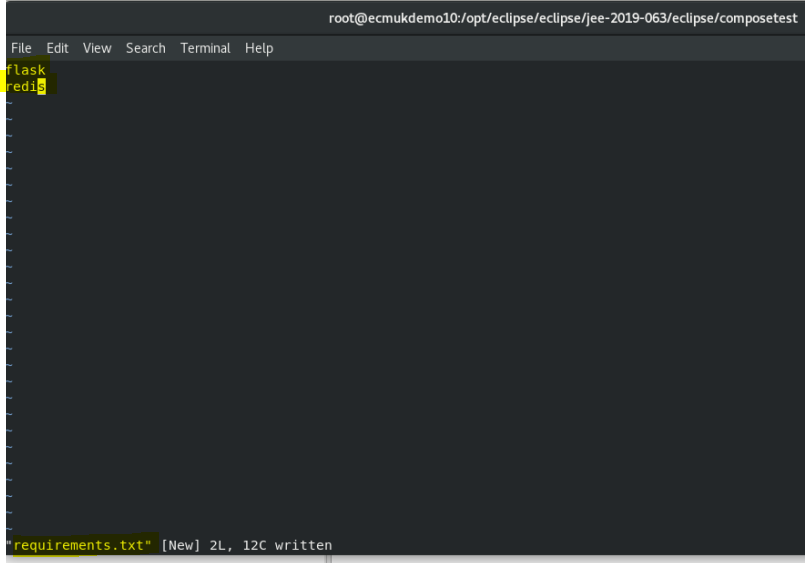
@app.route('/')

def hello():

count = get\_hit\_count()

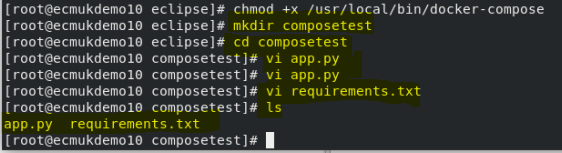
return 'Hello World! I have been seen {} times.\n'.format(count)

vi requirements.txt



flask

redis



**NOTE: yum install python37 - not able to install this, the existing version python36 is the latest for repos I have on RHEL 8.0 !**

## Create a Dockerfile

In this step, you write a Dockerfile that builds a Docker image. The image contains all the dependencies the Python application requires, including Python itself.

In your project directory, create a file named Dockerfile and paste the following:

FROM python:3.6-alpine

WORKDIR /code

ENV FLASK\_APP app.py

ENV FLASK\_RUN\_HOST 0.0.0.0

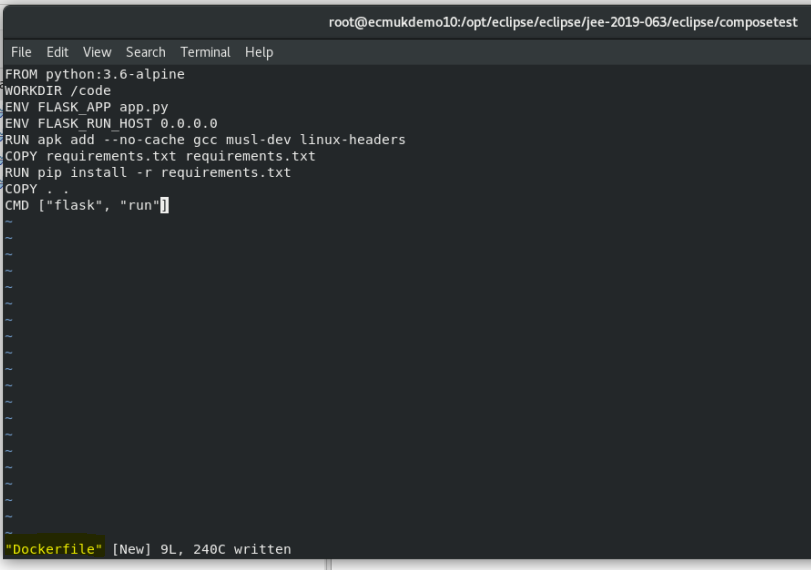
RUN apk add --no-cache gcc musl-dev linux-headers

COPY requirements.txt requirements.txt

RUN pip install -r requirements.txt

COPY . .

CMD ["flask", "run"]



This tells Docker to:

* Build an image starting with the **Python 3.6** image.
* Set the working directory to /code.
* Set environment variables used by the flask command.
* Install gcc so Python packages such as MarkupSafe and SQLAlchemy can compile speedups.
* Copy requirements.txt and install the Python dependencies.
* Copy the current directory . in the project to the workdir . in the image.
* Set the default command for the container to flask run.

Define services in a Compose file

Create a file called **docker-compose.yml** in your project directory and paste the following:

version: '3'

services:

web:

build: .

ports:

- "5000:5000"

redis:

image: "redis:alpine"

This Compose file defines two services: web and redis.

Web service

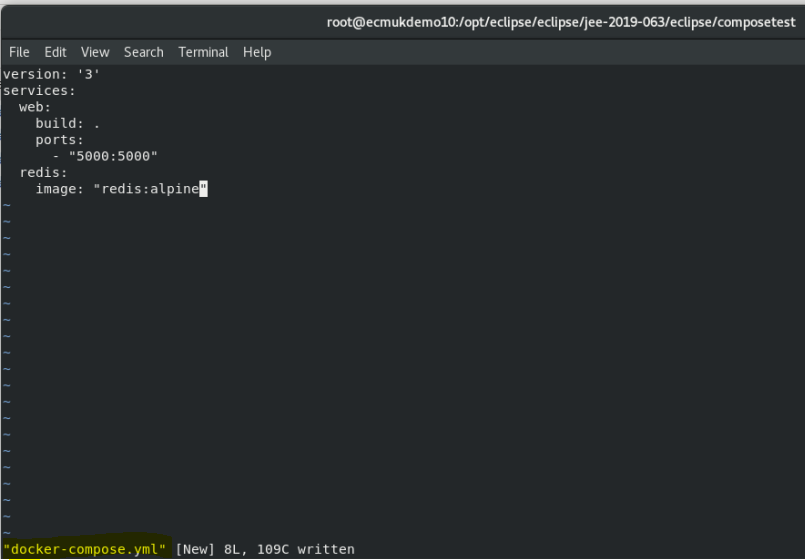
The web service uses an image that’s built from the Dockerfile in the current directory. It then binds the container and the host machine to the exposed port, 5000. This example service uses the default port for the Flask web server, 5000.

Redis service

The redis service uses a public Redis image pulled from the Docker Hub registry.

## Build and run your app with Compose

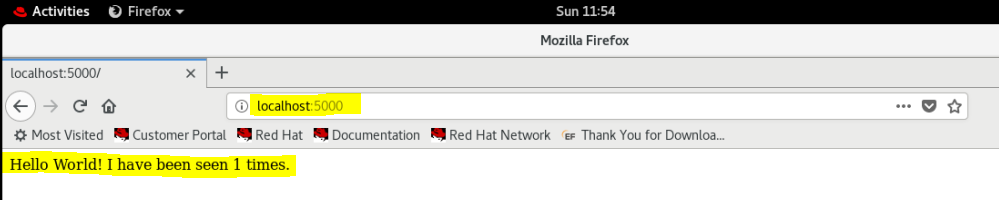
From your project directory, start up your application by running docker-compose up.



**docker-compose up**

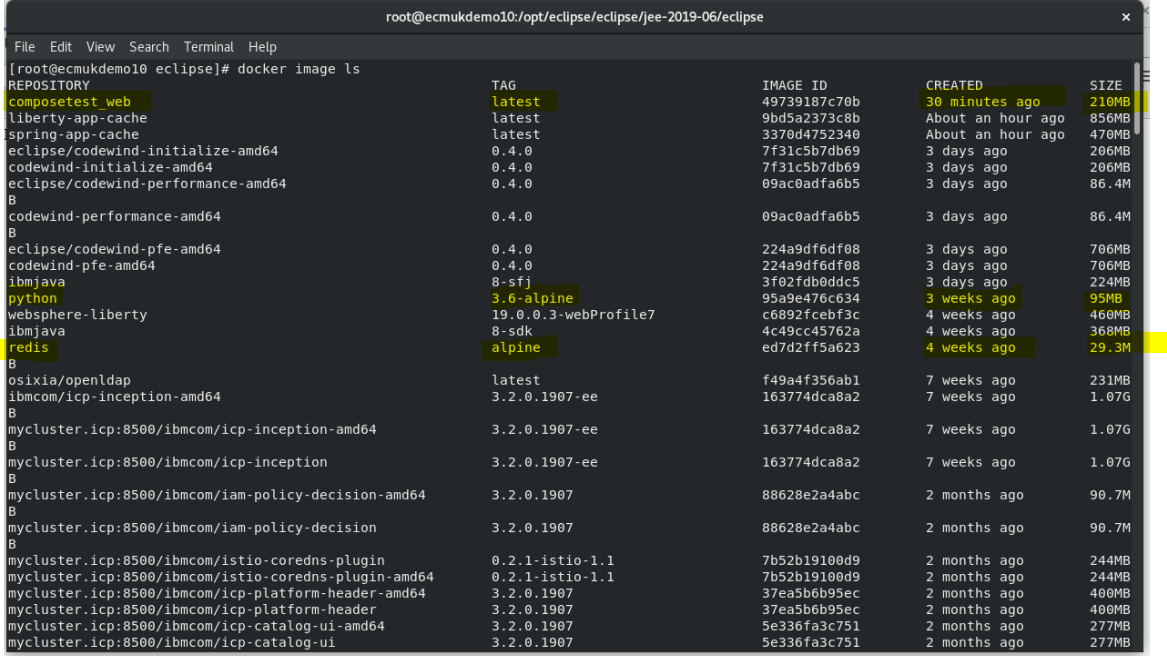
See Appendix A – Example Test Docker Container for the full compose log

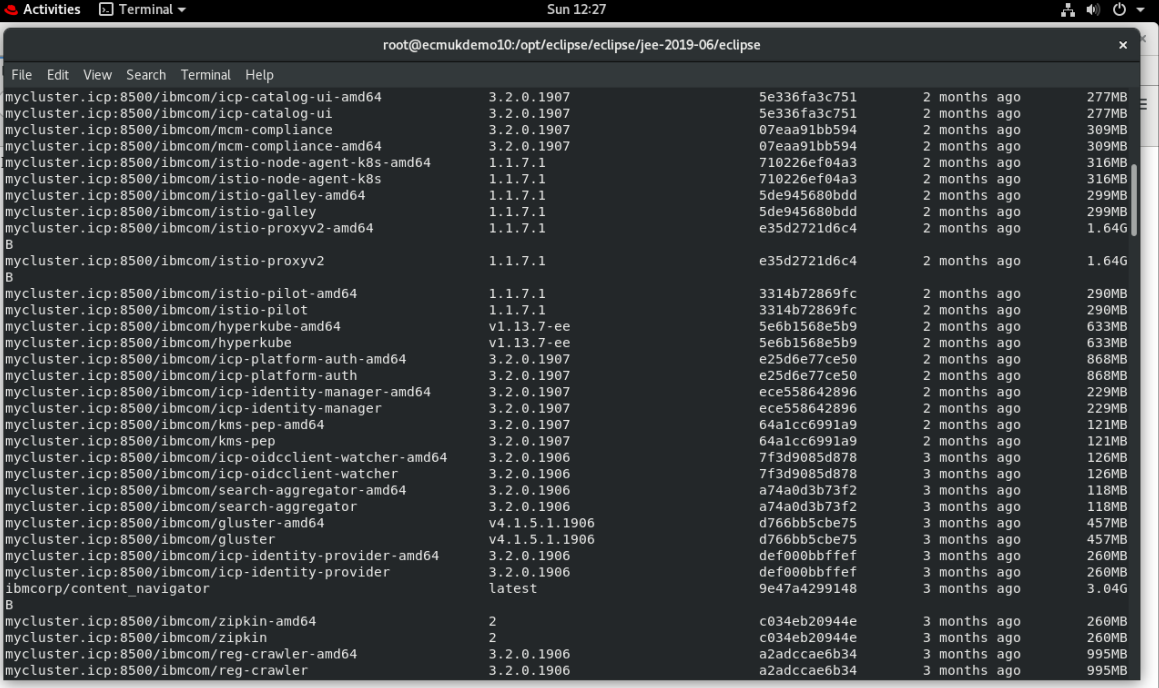
Enter http://localhost:5000/ in a browser to see the application running.

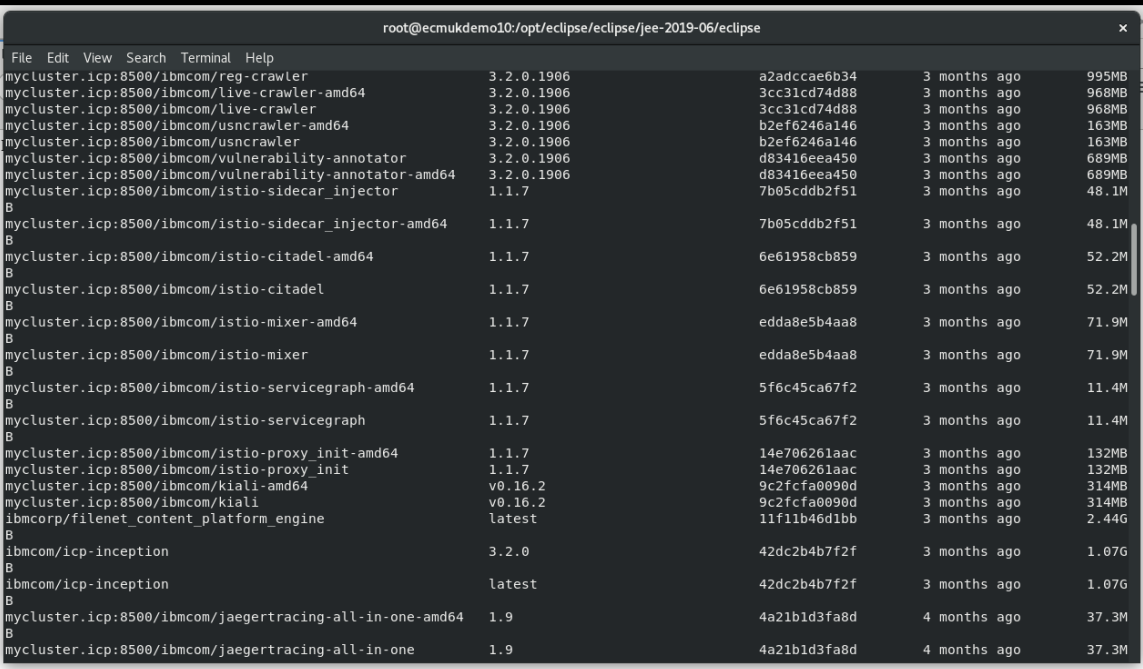


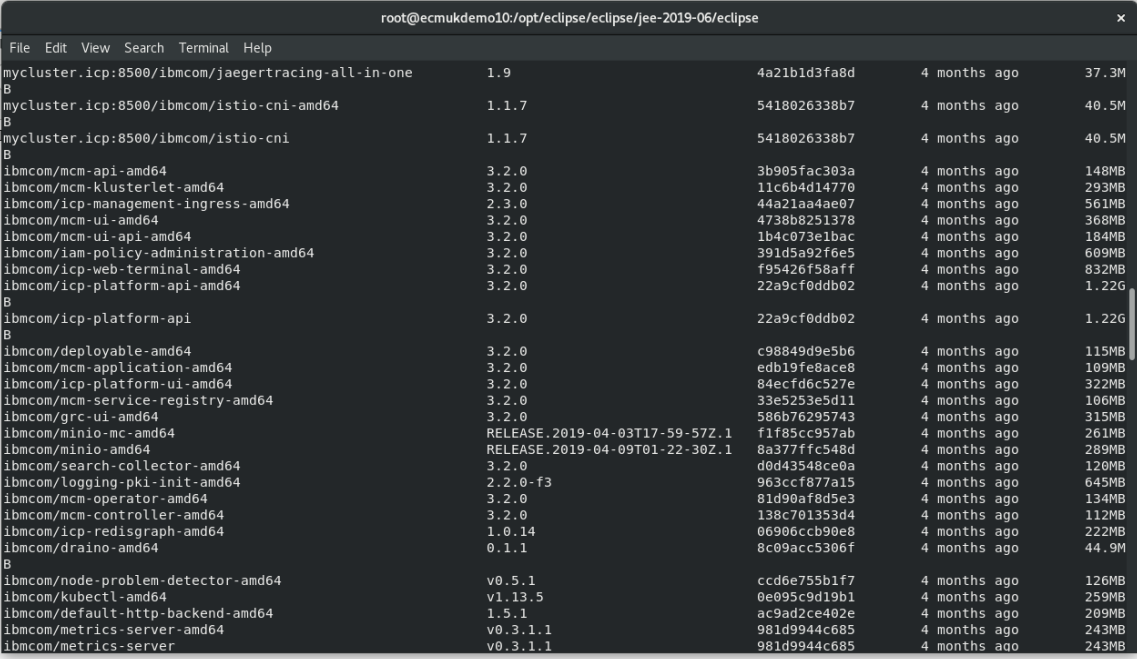
List the installed container images using

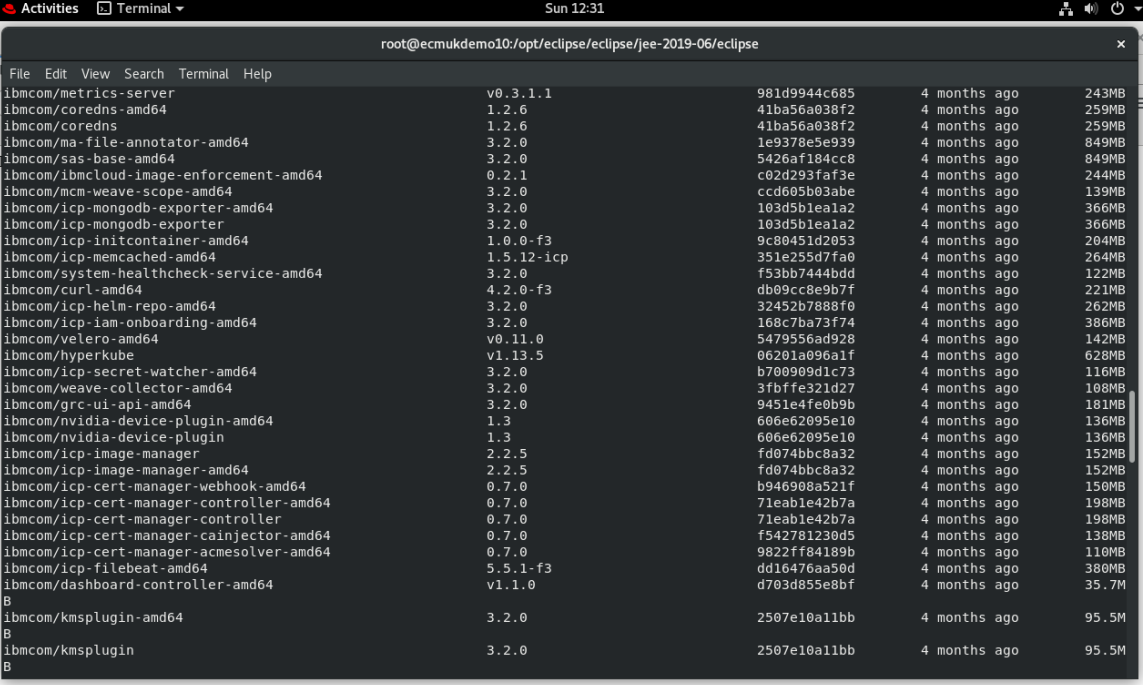
**docker image ls**

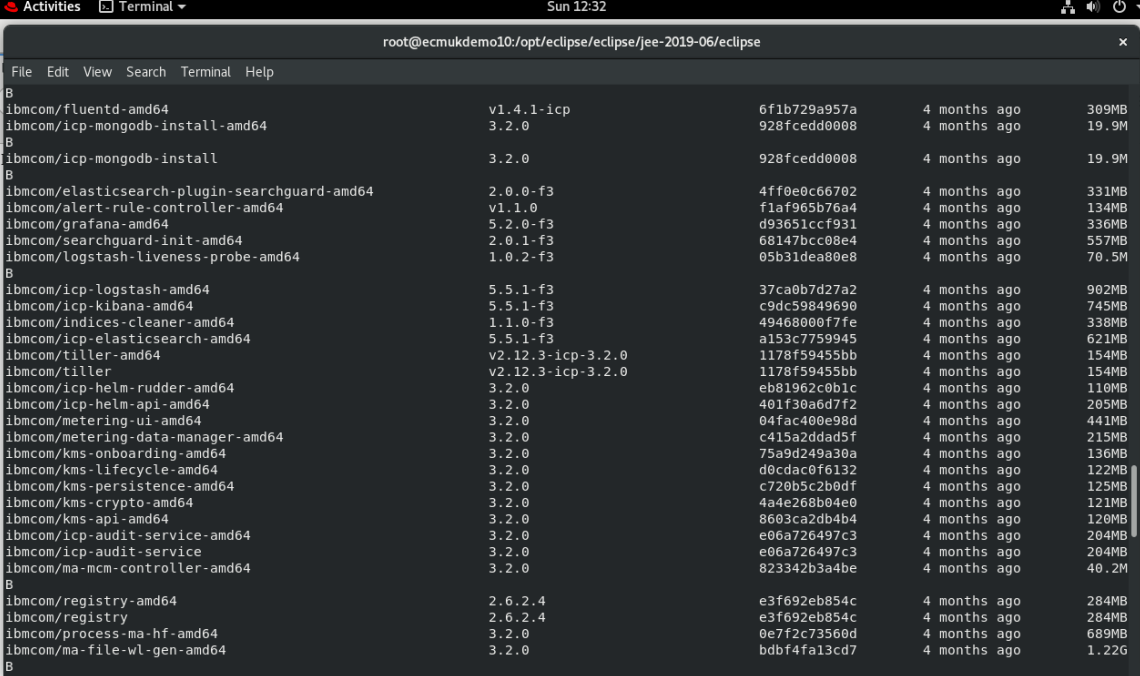


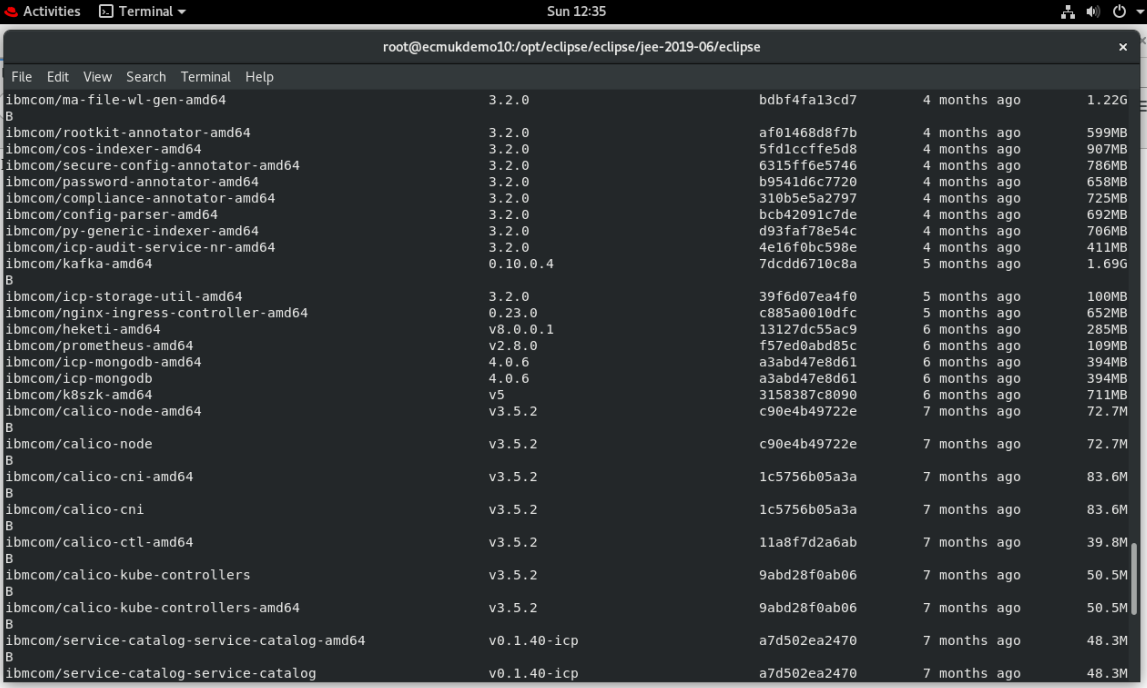


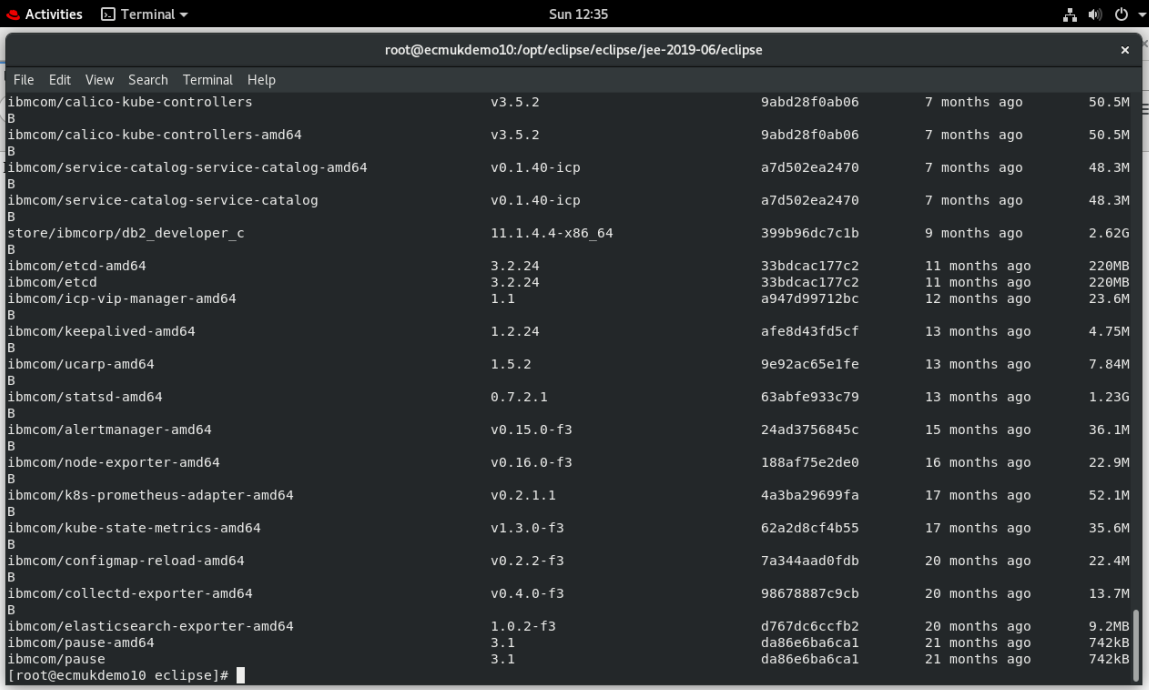




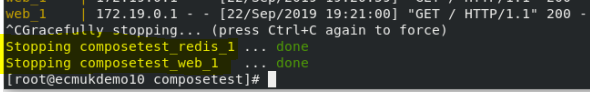








CTRL/C to stop the docker images loaded earlier



## Edit the Compose file to add a bind mount

Edit docker-compose.yml in the project directory to add a bind mount for the web service:

version: '3'

services:

web:

build: .

ports:

- "5000:5000"

**volumes:**

**- .:/code**

**environment:**

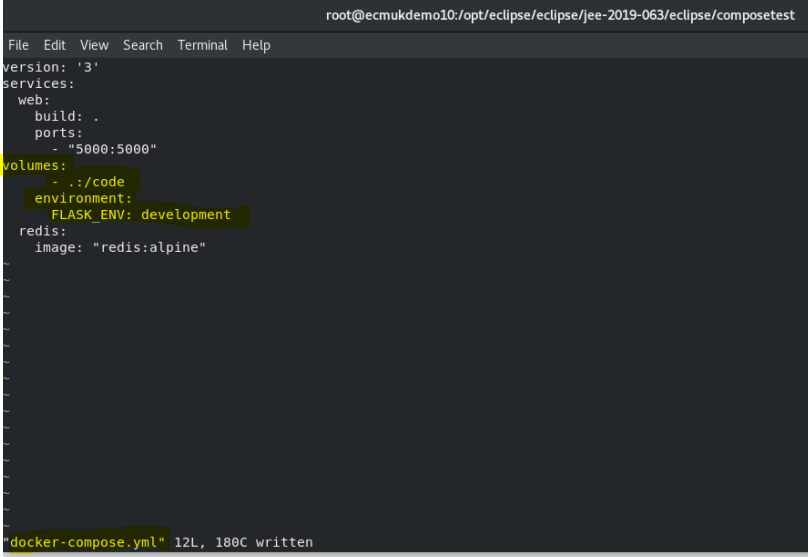
**FLASK\_ENV: development**

redis:

image: "redis:alpine"

The new volumes key mounts the project directory (current directory) on the host to /code inside the container, allowing you to modify the code on the fly, without having to rebuild the image. The environment key sets the FLASK\_ENV environment variable, which tells flask run to run in development mode and reload the code on change.

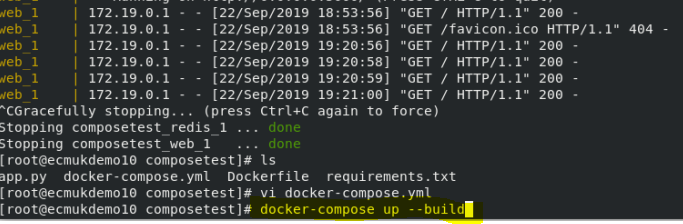
**NOTE:** **This mode should only be used in development.**

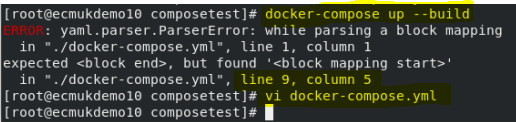


**NB – volumes : line above has to be indented !!!**

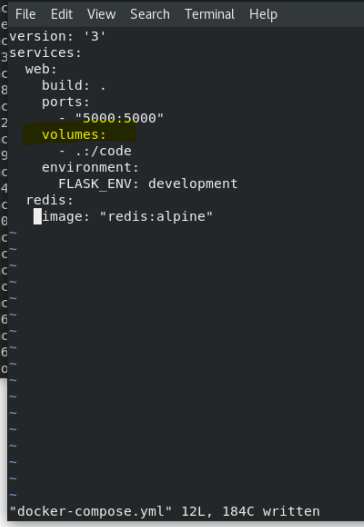
**(see below for what happens otherwise)**

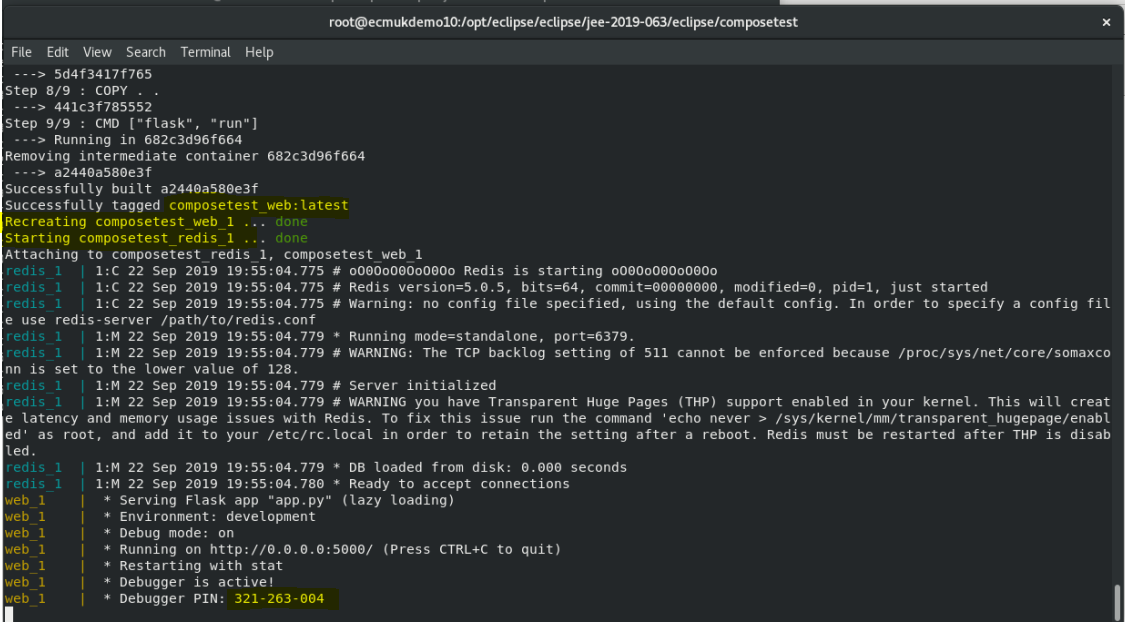
**docker-compose up –build**





**Indent the volumes: line!**

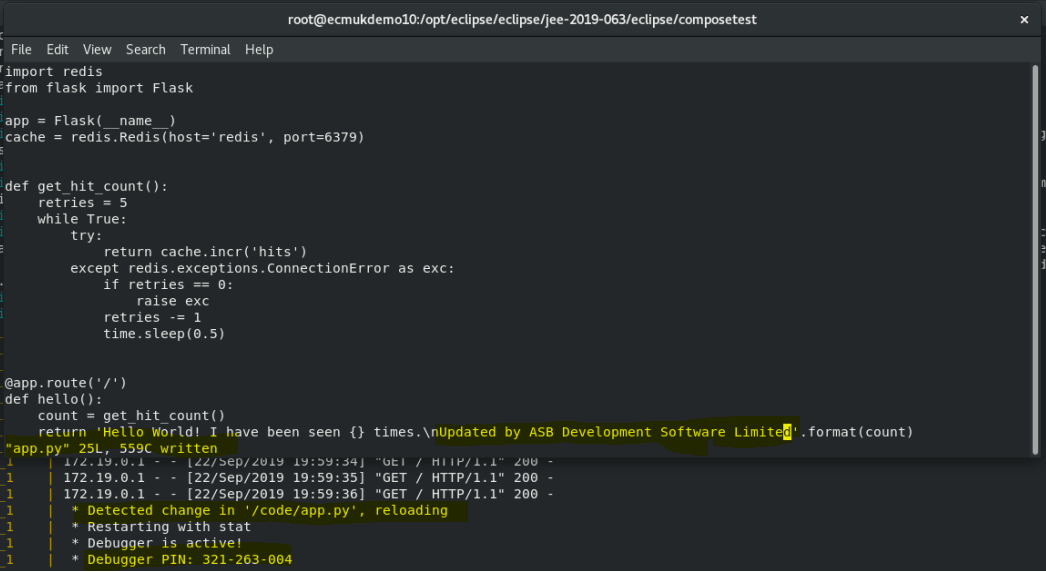


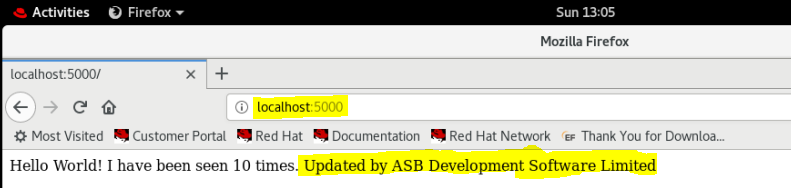


**Update** Change the greeting in app.py and save it. For example, change the Hello World! message to Hello from Docker!:

return 'Hello from Docker! I have been seen {} times.\nUpdated by ASB Development Software Limited'.format(count)

**vi app.py**





## Experiment with some other commands

If you want to run your services in the background, you can pass the -d flag (for “detached” mode) to docker-compose up and use docker-compose ps to see what is currently running:

# **docker-compose up -d**

Starting composetest\_redis\_1...

Starting composetest\_web\_1...

$ **docker-compose ps**

Name Command State Ports

-------------------------------------------------------------------

composetest\_redis\_1 /usr/local/bin/run Up

composetest\_web\_1 /bin/sh -c python app.py Up 5000->5000/tcp

The docker-compose run command allows you to run one-off commands for your services. For example, to see what environment variables are available to the web service:

# **docker-compose run web env**

See docker-compose --help to see other available commands. You can also install command completion for the bash and zsh shell, which also shows you available commands.

If you started Compose with docker-compose up -d, stop your services once you’ve finished with them:

# **docker-compose stop**

You can bring everything down, removing the containers entirely, with the down command. Pass --volumes to also remove the data volume used by the Redis container:

# **docker-compose down --volumes**

At this point, you have seen the basics of how Compose works.

# Appendix A – Example Test Docker Container

[root@ecmukdemo10 composetest]# **ls**

app.py Dockerfile requirements.txt

[root@ecmukdemo10 composetest]# **vi docker-compose.yml**

[root@ecmukdemo10 composetest]# **ls**

app.py docker-compose.yml Dockerfile requirements.txt

[root@ecmukdemo10 composetest]# **docker-compose up**

Creating network "composetest\_default" with the default driver

Building web

Step 1/9 : FROM python:3.6-alpine

3.6-alpine: Pulling from library/python

9d48c3bd43c5: Pull complete

c0ea575d71b9: Pull complete

c355bcdd89db: Pull complete

4738b6e326ab: Pull complete

99f07e28b502: Pull complete

Digest: sha256:40f1783b628264dfaeafdfff900b8b4375e5ed96a8d23620dc5cc6bb7d52148c

Status: Downloaded newer image for python:3.6-alpine

---> 95a9e476c634

Step 2/9 : WORKDIR /code

---> Running in 353d39f951a7

Removing intermediate container 353d39f951a7

---> b44f3a341e4e

Step 3/9 : ENV FLASK\_APP app.py

---> Running in acf004c6d1e0

Removing intermediate container acf004c6d1e0

---> de327ba27a82

Step 4/9 : ENV FLASK\_RUN\_HOST 0.0.0.0

---> Running in 6a10cc2a2ebf

Removing intermediate container 6a10cc2a2ebf

---> 1775405c3b09

Step 5/9 : RUN apk add --no-cache gcc musl-dev linux-headers

---> Running in 810ad76d6e36

fetch http://dl-cdn.alpinelinux.org/alpine/v3.10/main/x86\_64/APKINDEX.tar.gz

fetch http://dl-cdn.alpinelinux.org/alpine/v3.10/community/x86\_64/APKINDEX.tar.gz

(1/12) Installing binutils (2.32-r0)

(2/12) Installing gmp (6.1.2-r1)

(3/12) Installing isl (0.18-r0)

(4/12) Installing libgomp (8.3.0-r0)

(5/12) Installing libatomic (8.3.0-r0)

(6/12) Installing libgcc (8.3.0-r0)

(7/12) Installing mpfr3 (3.1.5-r1)

(8/12) Installing mpc1 (1.1.0-r0)

(9/12) Installing libstdc++ (8.3.0-r0)

(10/12) Installing gcc (8.3.0-r0)

(11/12) Installing linux-headers (4.19.36-r0)

(12/12) Installing musl-dev (1.1.22-r3)

Executing busybox-1.30.1-r2.trigger

OK: 121 MiB in 46 packages

Removing intermediate container 810ad76d6e36

---> 4bc7636af583

Step 6/9 : COPY requirements.txt requirements.txt

---> f94755f4113d

Step 7/9 : RUN pip install -r requirements.txt

---> Running in e70ca9408904

Collecting flask (from -r requirements.txt (line 1))

Downloading https://files.pythonhosted.org/packages/9b/93/628509b8d5dc749656a9641f4caf13540e2cdec85276964ff8f43bbb1d3b/Flask-1.1.1-py2.py3-none-any.whl (94kB)

Collecting redis (from -r requirements.txt (line 2))

Downloading https://files.pythonhosted.org/packages/bd/64/b1e90af9bf0c7f6ef55e46b81ab527b33b785824d65300bb65636534b530/redis-3.3.8-py2.py3-none-any.whl (66kB)

Collecting click>=5.1 (from flask->-r requirements.txt (line 1))

Downloading https://files.pythonhosted.org/packages/fa/37/45185cb5abbc30d7257104c434fe0b07e5a195a6847506c074527aa599ec/Click-7.0-py2.py3-none-any.whl (81kB)

Collecting itsdangerous>=0.24 (from flask->-r requirements.txt (line 1))

Downloading https://files.pythonhosted.org/packages/76/ae/44b03b253d6fade317f32c24d100b3b35c2239807046a4c953c7b89fa49e/itsdangerous-1.1.0-py2.py3-none-any.whl

Collecting Werkzeug>=0.15 (from flask->-r requirements.txt (line 1))

Downloading https://files.pythonhosted.org/packages/ce/42/3aeda98f96e85fd26180534d36570e4d18108d62ae36f87694b476b83d6f/Werkzeug-0.16.0-py2.py3-none-any.whl (327kB)

Collecting Jinja2>=2.10.1 (from flask->-r requirements.txt (line 1))

Downloading https://files.pythonhosted.org/packages/1d/e7/fd8b501e7a6dfe492a433deb7b9d833d39ca74916fa8bc63dd1a4947a671/Jinja2-2.10.1-py2.py3-none-any.whl (124kB)

Collecting MarkupSafe>=0.23 (from Jinja2>=2.10.1->flask->-r requirements.txt (line 1))

Downloading https://files.pythonhosted.org/packages/b9/2e/64db92e53b86efccfaea71321f597fa2e1b2bd3853d8ce658568f7a13094/MarkupSafe-1.1.1.tar.gz

Building wheels for collected packages: MarkupSafe

Building wheel for MarkupSafe (setup.py): started

Building wheel for MarkupSafe (setup.py): finished with status 'done'

Created wheel for MarkupSafe: filename=MarkupSafe-1.1.1-cp36-cp36m-linux\_x86\_64.whl size=31388 sha256=fec68302b18620dea6a6edeab84552c32f2253e959f660d7b16a7e0822639b58

Stored in directory: /root/.cache/pip/wheels/f2/aa/04/0edf07a1b8a5f5f1aed7580fffb69ce8972edc16a505916a77

Successfully built MarkupSafe

Installing collected packages: click, itsdangerous, Werkzeug, MarkupSafe, Jinja2, flask, redis

Successfully installed Jinja2-2.10.1 MarkupSafe-1.1.1 Werkzeug-0.16.0 click-7.0 flask-1.1.1 itsdangerous-1.1.0 redis-3.3.8

Removing intermediate container e70ca9408904

---> 5d4f3417f765

Step 8/9 : COPY . .

---> 4d834540a396

Step 9/9 : CMD ["flask", "run"]

---> Running in eff483e3a88f

Removing intermediate container eff483e3a88f

---> 49739187c70b

Successfully built 49739187c70b

Successfully tagged composetest\_web:latest

WARNING: Image for service web was built because it did not already exist. **To rebuild this image you must use `docker-compose build` or `docker-compose up --build`.**

Pulling redis (redis:alpine)...

alpine: Pulling from library/redis

9d48c3bd43c5: Already exists

6bcae78f4e99: Pull complete

8cb2d2938e96: Pull complete

8144cfccc349: Pull complete

4ff1d177b4d1: Pull complete

a67c684f8341: Pull complete

Digest: sha256:50899ea1ceed33fa03232f3ac57578a424faa1742c1ac9c7a7bdb95cdf19b858

Status: Downloaded newer image for redis:alpine

Creating composetest\_web\_1 ... **done**

Creating composetest\_redis\_1 ... **done**

Attaching to composetest\_redis\_1, composetest\_web\_1

redis\_1 | 1:C 22 Sep 2019 18:51:51.882 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

redis\_1 | 1:C 22 Sep 2019 18:51:51.882 # Redis version=5.0.5, bits=64, commit=00000000, modified=0, pid=1, just started

redis\_1 | 1:C 22 Sep 2019 18:51:51.882 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf

redis\_1 | 1:M 22 Sep 2019 18:51:51.884 \* Running mode=standalone, port=6379.

redis\_1 | 1:M 22 Sep 2019 18:51:51.884 # **WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.**

redis\_1 | 1:M 22 Sep 2019 18:51:51.884 # Server initialized

redis\_1 | 1:M 22 Sep 2019 18:51:51.884 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command **'echo never > /sys/kernel/mm/transparent\_hugepage/enabled**' as root, and add it to your **/etc/rc.local** in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

redis\_1 | 1:M 22 Sep 2019 18:51:51.885 **\* Ready to accept connections**

web\_1 | \* Serving Flask app "app.py"

web\_1 | \* Environment: production

web\_1 | **WARNING: This is a development server. Do not use it in a production deployment.**

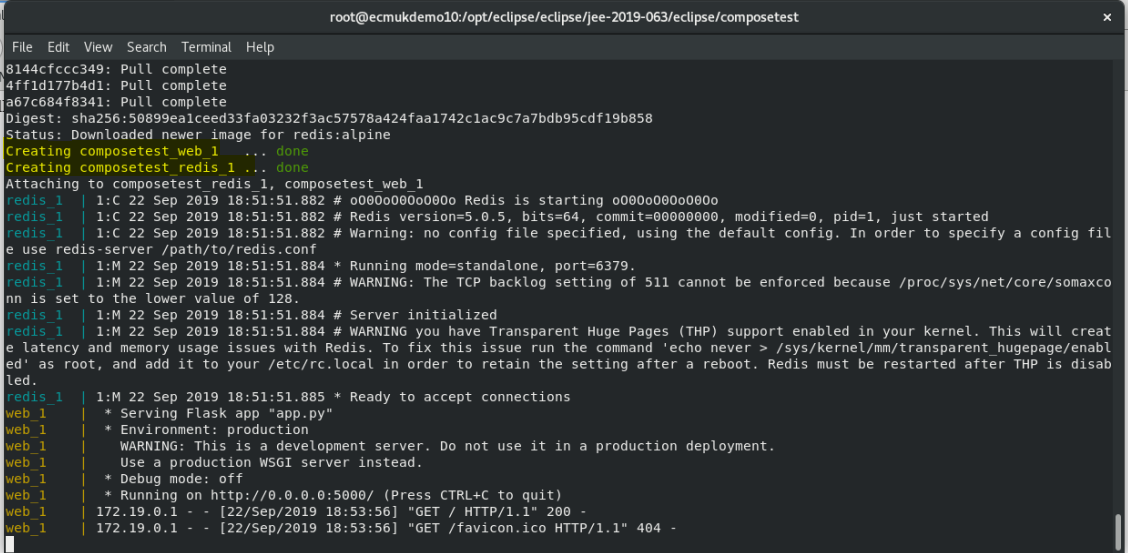
web\_1 | Use a production WSGI server instead.

web\_1 | \* Debug mode: off

web\_1 | \* Running on **http://0.0.0.0:5000/** (Press CTRL+C to quit)

web\_1 | 172.19.0.1 - - [22/Sep/2019 18:53:56] "GET / HTTP/1.1" 200 -

web\_1 | 172.19.0.1 - - [22/Sep/2019 18:53:56] "GET /favicon.ico HTTP/1.1" 404 -



# Appendix B - Example Eclipse plugin python container log

Running development environment...

\*

\*

\*

A new CLI update is available.

Please go to https://github.com/appsody/appsody/releases/latest and update from 0.4.3 --> 0.4.5.

\*

\*

\*

Running command: docker pull appsody/python-flask:0.1

Running docker command: docker run --rm -P --name cw-asbcodewindtest-63660d70-dd77-11e9-8727-53a434350696 --network codewind\_network -v /root/codewind-workspace/ASBCodeWindTest/:/project/userapp -v asbcodewindtest-deps:/project/deps -v /root/codewind-workspace/.extensions/codewind-appsody-extension/bin/appsody-controller:/appsody/appsody-controller -t --entrypoint /appsody/appsody-controller appsody/python-flask:0.1 --mode=run

[Container] Running Install: cd /project/userapp;pipenv lock -r > requirements.txt;python -m pip install -r requirements.txt -t /project/deps

[Container] Creating a virtualenv for this project…

[Container] Pipfile: /project/userapp/Pipfile

[Container] Using /usr/local/bin/python (3.7.4) to create virtualenv…

[Container] ?25l

⠋0m Creating virtual environment...K

⠋0m Creating virtual environment...K

Etc….

⠙0m Creating virtual environment...K

Already using interpreter /usr/local/bin/python

[Container] Using base prefix '/usr/local'

[Container] New python executable in /root/.local/share/virtualenvs/userapp-pHZHaS8v/bin/python

[Container] Installing setuptools, pip, wheel...

[Container] done.

[Container]

[Container]

K?25h32m22m✔ Successfully created virtual environment!39m22m 0m

[Container] Virtualenv location: /root/.local/share/virtualenvs/userapp-pHZHaS8v

[Container] Pipfile.lock not found, creating…

[Container] Locking dev-packages dependencies…

[Container] Locking packages dependencies…

[Container] Updated Pipfile.lock (a65489)!

[Container] Running: python -m flask run --host=0.0.0.0 --port=8080

[Container] \* Serving Flask app "server"

[Container] \* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)

[Container] 172.18.0.3 - - 22/Sep/2019 20:27:43 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 20:27:53 "GET /health HTTP/1.1" 200 -

**Etc… etc..**

[Container] 172.18.0.3 - - 22/Sep/2019 21:20:53 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:03 "GET /health HTTP/1.1" 200 -

**[Error] Error waiting in 'appsody run' exit status 137**

Running development environment...

\*

\*

\*

A new CLI update is available.

**Please go to**

**https://github.com/appsody/appsody/releases/latest and update from 0.4.3 --> 0.4.5**.

\*

\*

\*

Running command: docker pull appsody/python-flask:0.1

Running docker command: docker run --rm -P --name cw-asbcodewindtest-63660d70-dd77-11e9-8727-53a434350696 --network codewind\_network -v /root/codewind-workspace/ASBCodeWindTest/:/project/userapp -v asbcodewindtest-deps:/project/deps -v /root/codewind-workspace/.extensions/codewind-appsody-extension/bin/appsody-controller:/appsody/appsody-controller -t --entrypoint /appsody/appsody-controller appsody/python-flask:0.1 --mode=run

[Container] Running Install: cd /project/userapp;pipenv lock -r > requirements.txt;python -m pip install -r requirements.txt -t /project/deps

**[Container] Warning: Python 3.6 was not found on your system…**

[Container] You can specify specific versions of Python with:

[Container] $ pipenv --python path/to/python

[Container] Running: python -m flask run --host=0.0.0.0 --port=8080

[Container] \* Serving Flask app "server"

[Container] \* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:17 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:19 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:21 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:23 "GET /health HTTP/1.1" 200 -

Stopping development environment

Running command: docker stop cw-asbcodewindtest-63660d70-dd77-11e9-8727-53a434350696

Closing down development environment, sleeping 60 seconds: exit status 2

Running development environment...

\*

\*

\*

A new CLI update is available.

Please go to https://github.com/appsody/appsody/releases/latest and update from 0.4.3 --> 0.4.5.

\*

\*

\*

Running command: docker pull appsody/python-flask:0.1

Running docker command: docker run --rm -P --name cw-asbcodewindtest-63660d70-dd77-11e9-8727-53a434350696 --network codewind\_network -v /root/codewind-workspace/ASBCodeWindTest/:/project/userapp -v asbcodewindtest-deps:/project/deps -v /root/codewind-workspace/.extensions/codewind-appsody-extension/bin/appsody-controller:/appsody/appsody-controller -t --entrypoint /appsody/appsody-controller appsody/python-flask:0.1 --mode=run

[Container] Running Install: cd /project/userapp;pipenv lock -r > requirements.txt;python -m pip install -r requirements.txt -t /project/deps

[Container] Warning: Python 3.6 was not found on your system…

[Container] You can specify specific versions of Python with:

[Container] $ pipenv --python path/to/python

[Container] Running: python -m flask run --host=0.0.0.0 --port=8080

[Container] \* Serving Flask app "server"

[Container] \* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:37 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:39 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:21:41 "GET /health HTTP/1.1" 200 -

Etc… etc

[Container] 172.18.0.3 - - 22/Sep/2019 21:25:53 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:26:13 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:26:33 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:26:53 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:27:13 "GET /health HTTP/1.1" 200 -

[Container] 172.18.0.3 - - 22/Sep/2019 21:27:33 "GET /health HTTP/1.1" 200 –

[Container] 172.18.0.3 - - 22/Sep/2019 21:31:53 "GET /health HTTP/1.1" 200 -

**Closing down development environment, sleeping 60 seconds: exit status 2**