Rocket® software

Rocket® UniVerse Docker How to

Contents

Overview

This document demonstrates how to install and run Rocket® UniVerse in a Docker container. Running UniVerse in a Container is recommended for non-production environments at this time. Containers want all dynamic information such as logs or database items to be outside the container. Currently UniVerse mixes these items in the UVHOME directory and therefore violates these rules. For non-production environments this will be less of an issue.

Sample Project

The sample project is a Docker Compose project with a single UniVerse Container. It has all the required scripts to setup and configure a Rocky Linux Container and install UniVerse. You must go to RBC and obtain the version of UniVerse you wish to use and drop it into the project under the shared-data/universe-install-file directory. Licensing must also be obtained to fully activate UniVerse.

Note on networking

This docker will expose UniRPC via port 31439. If you have these ports in use on your host, you will have to change these. For example, if you have UniVerse for Windows installed on your workstation you will need to change the docker-compose.yml file to say: “31439:31438” on line 11. When you use UniObjects you will now say to look for the Docker version at localhost:31439. The first number is what to listen on the host and the second number is where the service is really listening inside the container.

← Back to contents

Step 1

Install Docker Desktop

Install Docker Desktop. You can retrieve it from: https://www.docker.com/products/docker-desktop/

Step 2

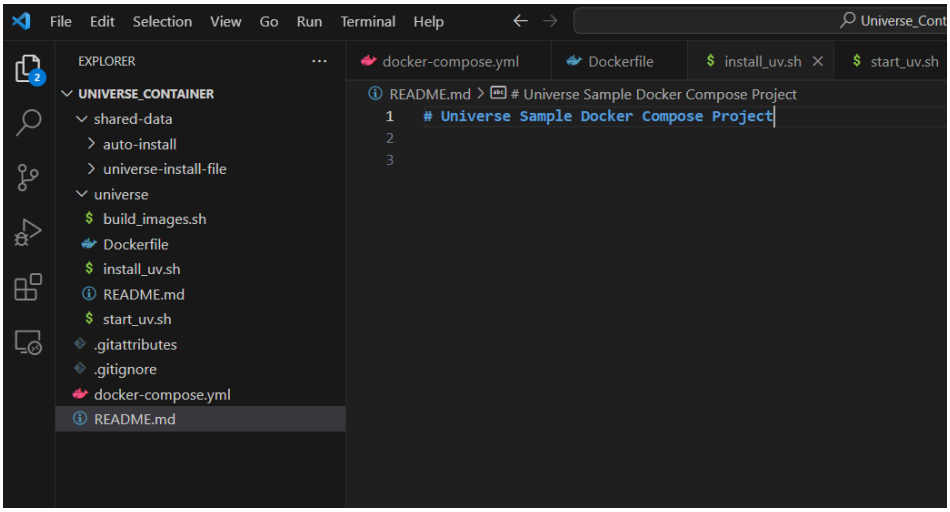
Install Visual Studio Code

Install Visual Studio Code. This is optional but if you wish to mirror these instructions then it is recommended. https://code.visualstudio.com/

Step 3

Download sample UniVerse Compose Project

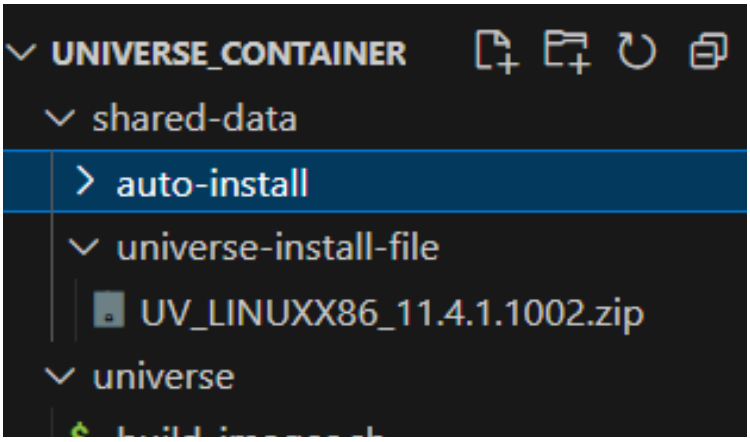
Download the sample project from https://github.com/RocketSoftware/multivalue-containers and install it on your machine. In this example it is in the users download area. Open the directory with Visual Studio Code.



Step 4

Retrieve UniVerse install Zip file from RBC

Retrieve the UniVerse install Zip file from RBC and place it in the shared-data/universe-install-file directory. Only have a single install file in this directory. The installer will scan the directory and automatically grab the file. If you put multiple copies in the directory, it will grab the last one it sees which can be confusing.

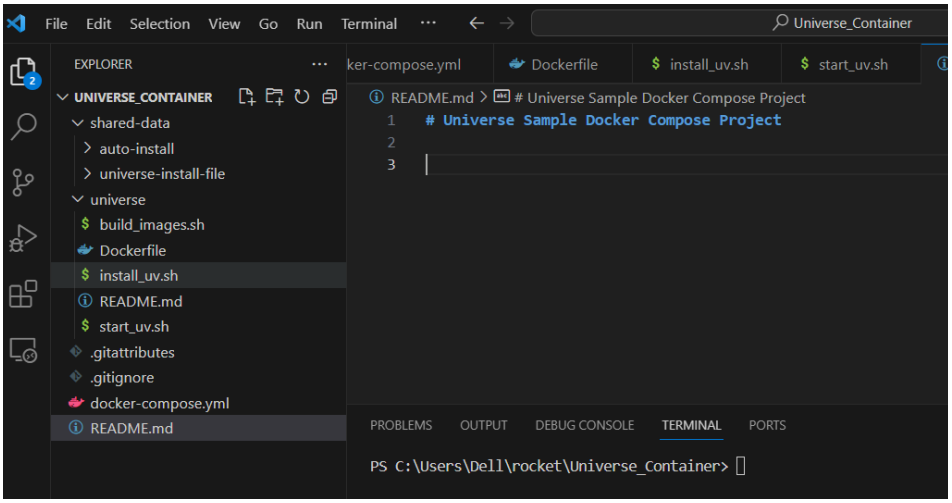


← Back to contents

Step 5

Open up Visual Studio Code

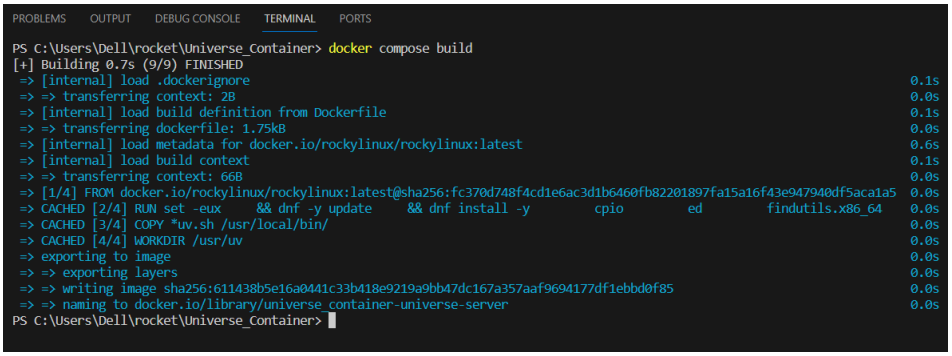
Open a terminal window in Visual Studio Code. It is the Terminal option across the top, choose new Terminal. It should open a new window with a command prompt.



Step 6

Build UniVerse Docker

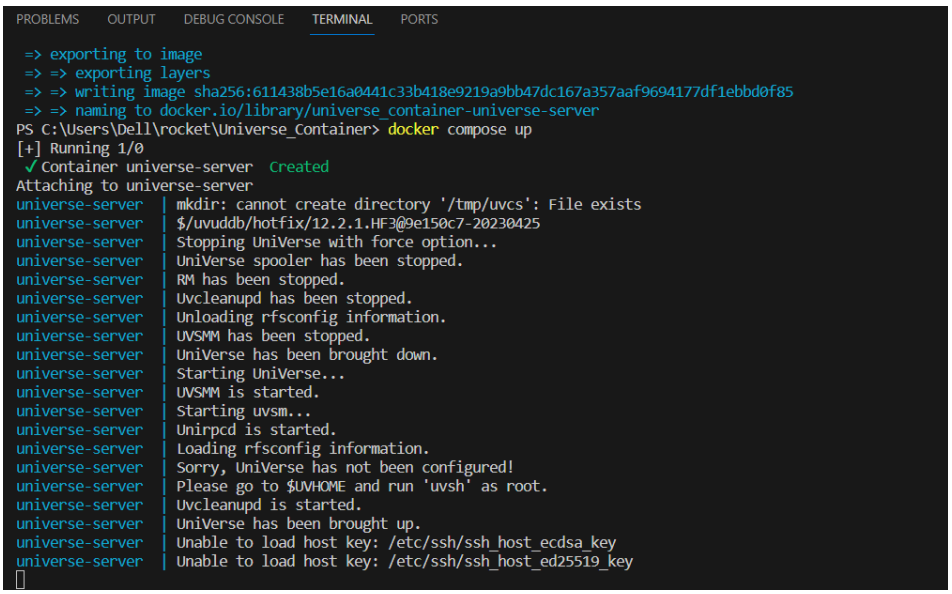
Type “docker compose build” in the terminal window. This will build the container using the sample Docker-compose yaml file and the UniVerse DockerFile and scripts.



Step 7

Start UniVerse Container and Install UniVerse

Type “docker compose up” in the terminal window. This will start up your UniVerse container and complete the installation. This step will take a while the first time since it is fully installing UniVerse. The below example is showing an already installed UniVerse starting up. The first time you will see much more information.

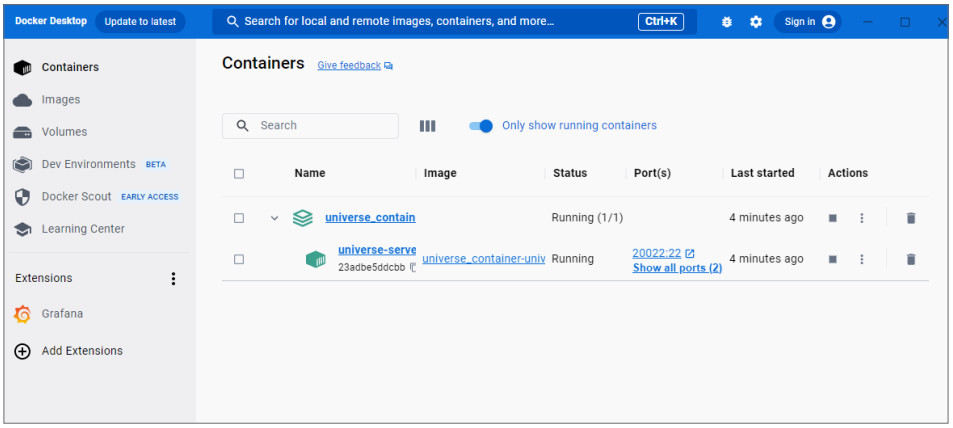


← Back to contents

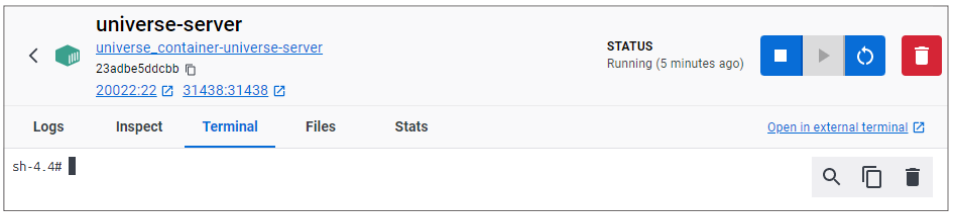
Step 8

Activate UniVerse

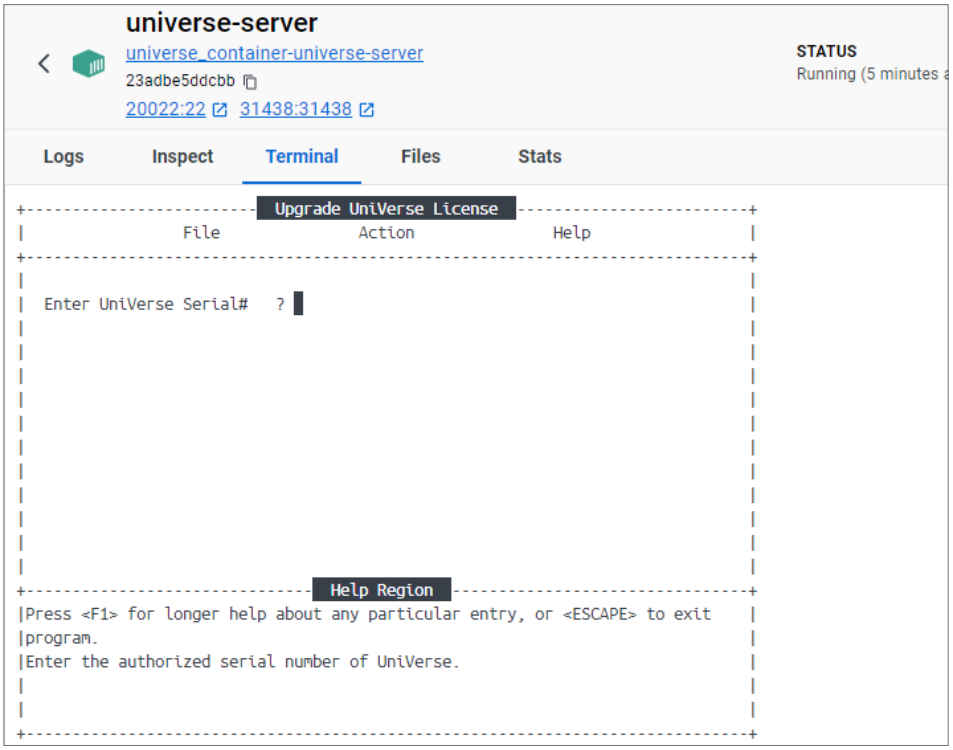
Open Docker Desktop. Click on the Containers tab. You should see your UniVerse Container running.



Click on the actual UniVerse-Server container (the lower one). Then click on Terminal to open up a bash prompt in the UniVerse container.

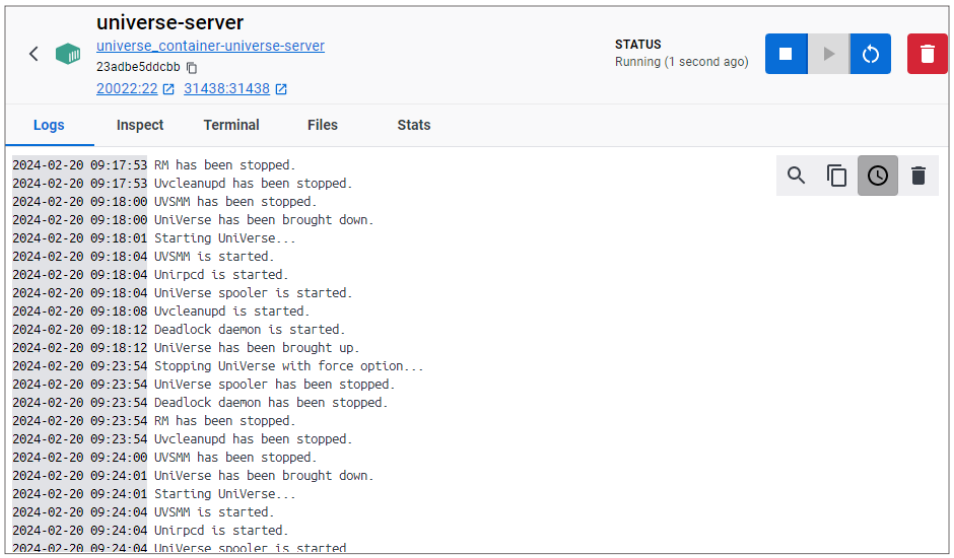


cd to the /usr/uv directory.
Then type uv. This should bring up the licensing page.



Activate UniVerse as you normally would.

Once activated restart UniVerse. This can be done by going back to the Docker desktop and changing from Terminal to Logs. Click the blue restart icon on the top right (it looks like a circle with an arrow). You will see the Docker container stop and restart and output logs. The new licensing will be active. You can now go back into the terminal window, cd to /usr/uv and then type uv to launch into the uv Admin account.

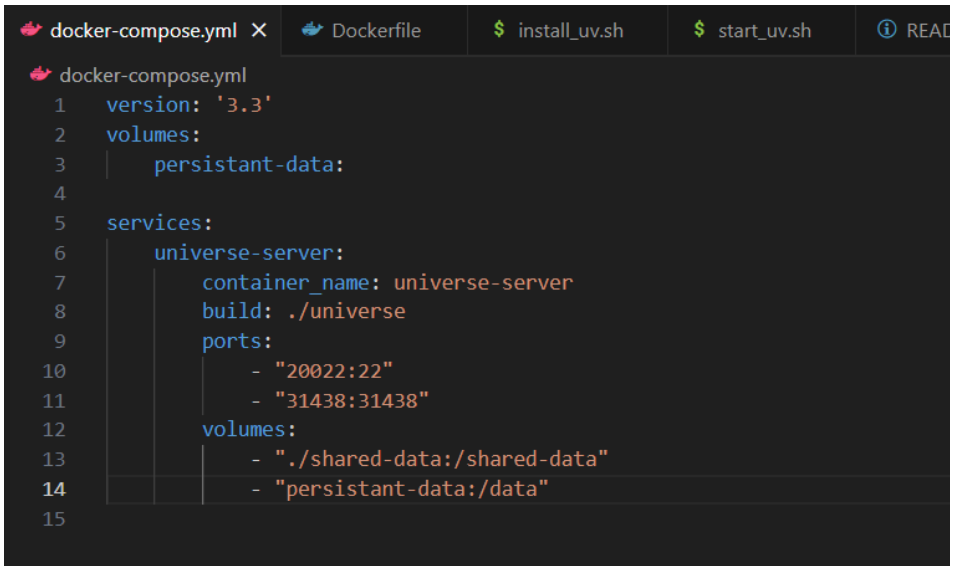


Customization and Best Practices

Data Storage

We recommend that you do all account work outside the container in mounted storage. This container creates a Storage Volume called persistent-data and is mounted under /data in the container. Any accounts you create should be in this directory. A Storage Volume is your own personal mounted storage. It is independent from the Container and will survive even if you delete the container.

The storage is defined in the docker-compose.yml file.



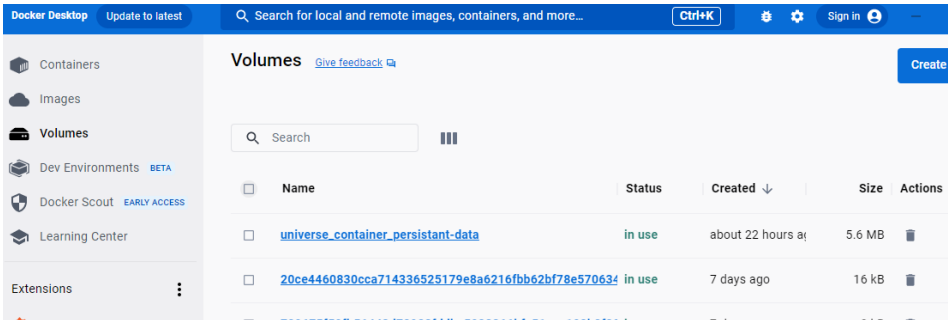
The volume and its name is defined in the volumes section (see line 3) The storage is mounted in the universe-server section under volumes. See line 14.

← Back to contents

This storage area is not in the container project (as opposed to the shared-data link) and will not therefore be tracked in git if the project is setup with git, nor will information be tracked by Docker.

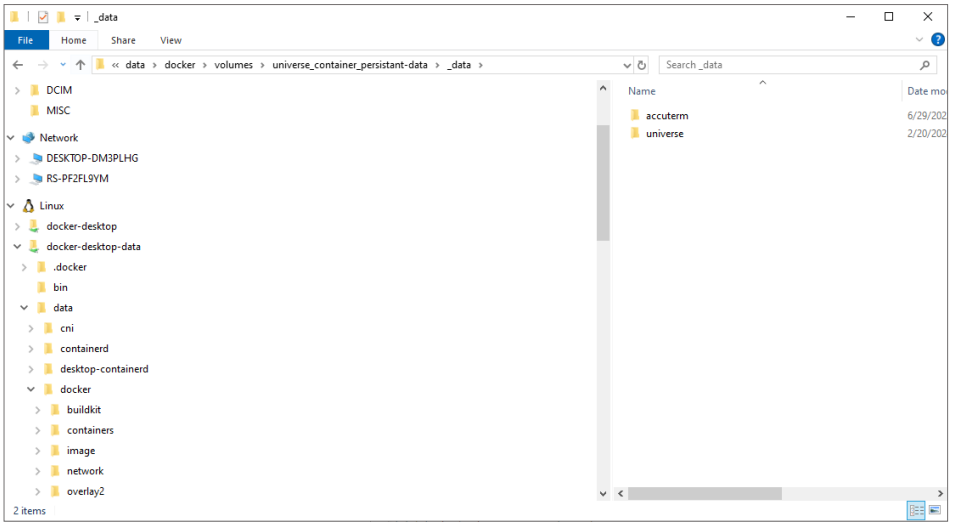
You can also view this storage in Docker Desktop under the Volumes Tab.

When you click on the storage container you will get a summary tab of the storage and any containers currently using this storage.



The data tab will show you what data is in the storage.

Advanced Access: Based on your platform and how you installed Docker (recommendation is WSL 2) you can directly access your storage and you can find your storage here (see diagram above).



← Back to contents

Automatic Licensing

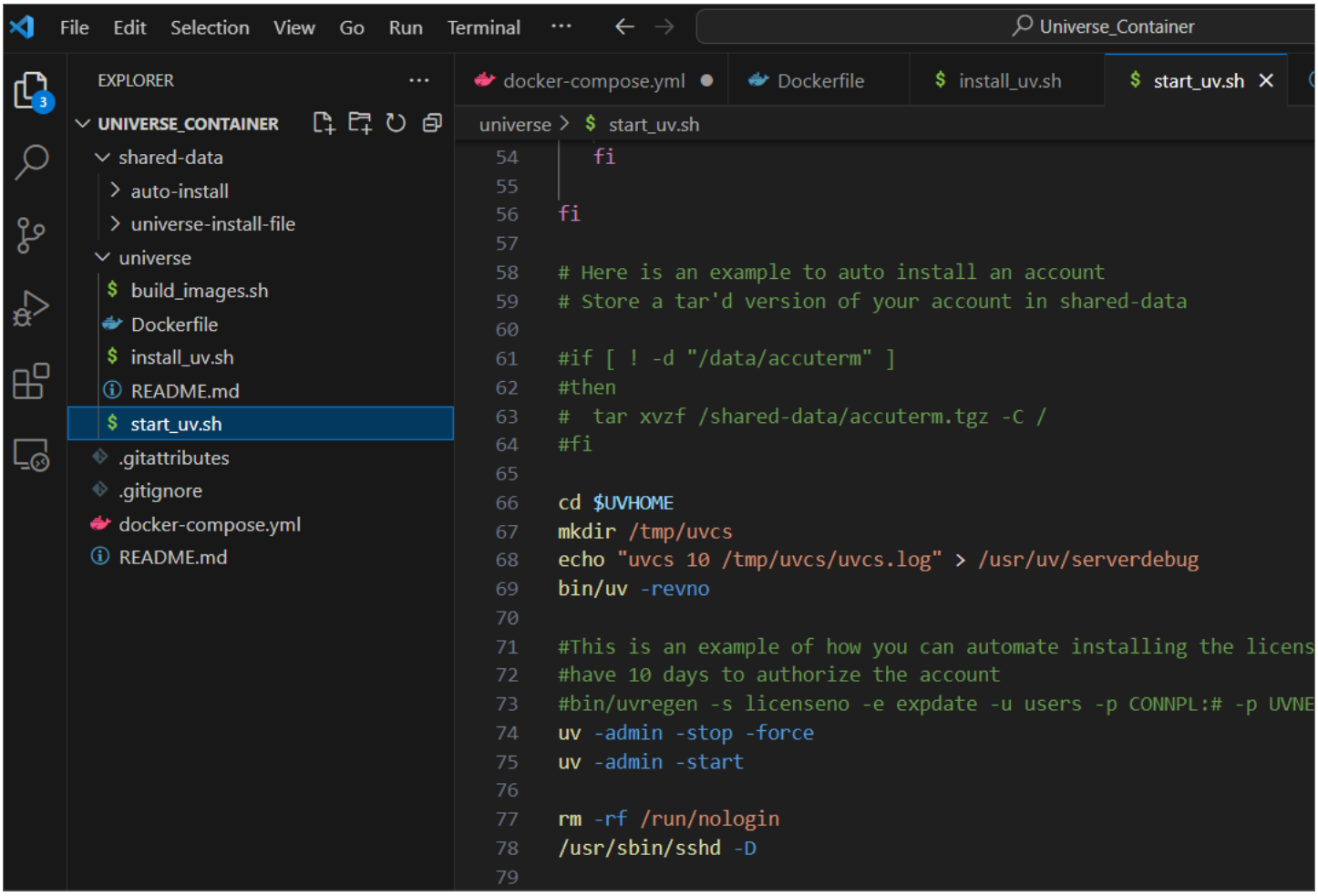
You can automate the licensing using the uvregen tool. Open Visual Studio Code, open the universe folder and open the start_uv.sh file. Scroll down to the bottom of the file.

From the terminal:

1. Stop the container (ctrl-c).
2. Docker compose rm (this will remove your current container and anything you have done inside the container will be lost).
3. Docker compose build (this will rebuild the project with your updated start_uv.sh script).
4. Docker compose up.

If you do not wish to lose your UniVerse container you can also:

1. cd to /usr/local/bin in the Docker Container.
2. Update the start_uv.sh script. Update line 73.
3. You can now just restart the container using the Docker desktop restart button.



← Back to contents

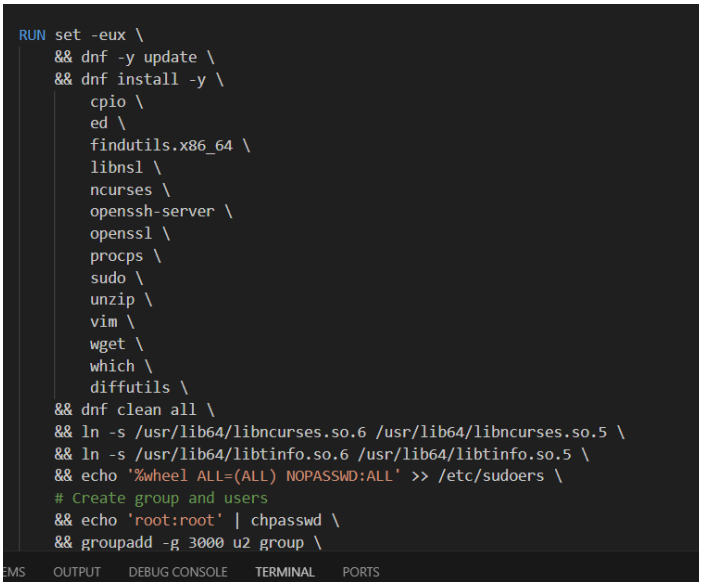
Customizing the Install

You can make modifications to how UniVerse is setup and configured. You'll make most of these modifications in Dockerfile (new install dependencies) and start_uv.sh.

New Dependencies

If you wish to install new dependencies, modify the DockerFile in the UniVerse directory.

At the top you will see it using dnf to install packages. If you wish to auto-install other tools such as git first do it within the running container. You can then add it here. Make sure if you add a new entry, you do it after diffutils and make sure you add the ending \ (this tells it to add all these as one large command script).

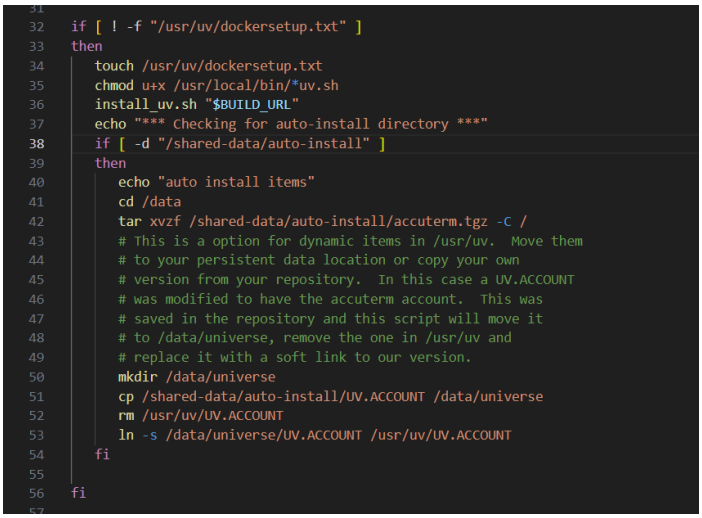


Auto Install an Account

Look at the start_uv.sh script for examples on how to conditionally add items. These conditions keep the container from repeating these steps.

When it is first configured, the container looks for a text file called /usr/uv/dockersetup.txt. This file is created by this script (see line 34) and therefore will only do these steps on a new container.

The second if question looks to see if /shared-data/auto-install exists. If it does it unpacks an accuterm.tgz backup file. This same directory is in the project under shared-data and is the same directory. This is an easy way to copy items into your container. Keep in mind the AccuTerm account does not exist in the UV.ACCOUNT file at this time.



← Back to contents

Custom Items from UVHOME

You can handle some dynamic items from UVHOME by either relocating them to your persistent storage area and setting a sym link or just manually back items up and copy them in.

In this example we are going to use an updated UV.ACCOUNT file that has our ACCUTERM account already in it.

```
49      # replace it with a soft link to our version.
50      mkdir /data/universe
51      cp /shared-data/auto-install/UV.ACCOUNT /data/universe
52      rm /usr/uv/UV.ACCOUNT
53      ln -s /data/universe/UV.ACCOUNT /usr/uv/UV.ACCOUNT
54    fi
55
```

1. Create a directory in our persistent area (we called it universe). Again, this is your personal Volume and will persist even if containers are destroyed.
2. Copy our updated UV.ACCOUNT from our project (/shared-data/auto-install/UV.ACCOUNT) to /data/universe.
3. Remove the one installed by the UniVerse Installer.
4. Set a soft link in the original /usr/uv/UV.ACCOUNT location and point it to the one in the data mount.
5. Build a container; now it will now run these steps.
6. Keep in mind you can do these items manually inside the container until you have your steps good.

In the above example you can now rebuild a container at any time and any work you do in UV.ACCOUNTS will stay. When you create new accounts you should be creating them in /data. This means both your accounts and your UV.ACCOUNTS definition of those accounts will hold even if you delete and re-create the container.

Other items inside UVHOME

1. The global catalog directory. If you do not relocate this directory, you will have to recompile/catalog any globally cataloged items. If you are doing local catalog work, you will not have an issue.
2. Configuration changes (audit logging, uv.config, etc).
3. Replication (the logs are often in UVHOME). This is configurable and it's best to relocate to your persistent location.

← Back to contents

Advanced Items

Other Containers

You can add other containers to this project. All you need to do is add additional containers to the services section of the docker-compose.yml file. If you are doing Docker build modifications to the container you would also create a new subdirectory to contain the custom DockerFile and build items. Most Docker projects have examples.

Networking

By default, this project exposes ssh and UniObjects to the host machine.

- ssh – localhost 20022
- UniObjects – localhost 31438

If you have these ports in use on your host, you will have to change these. For example, if you have UniVerse for Windows installed on your workstation you will need to change the docker-compose.yml file to say: “31439:31438” on line 11. When you use UniObjects you will now say to look for the Docker version at localhost:31439. The first number is what to listen on the host and the second number is where the service is really listening inside the container.

If you have multiple containers running in the same project, you can ping each one via the hostname. Docker compose does this dns work for you automatically.

About Rocket Software

Rocket Software partners with the largest Fortune 1000 organizations to solve their most complex IT challenges across Applications, Data and Infrastructure. Rocket Software brings customers from where they are in their modernization journey to where they want to be by architecting innovative solutions that deliver next-generation experiences. Over 10 million global IT and business professionals trust Rocket Software to deliver solutions that improve responsiveness to change and optimize workloads. Rocket Software enables organizations to modernize in place with a hybrid cloud strategy to protect investment, decrease risk and reduce time to value. Rocket Software is a privately held U.S. corporation headquartered in the Boston area with centers of excellence strategically located throughout North America, Europe, Asia and Australia. Rocket Software is a portfolio company of Bain Capital Private Equity. Follow Rocket Software on LinkedIn and X (formerly Twitter).

Rocket® software

Modernization. Without Disruption.™

Learn more

Visit RocketSoftware.com ›

MAR-10149_WP_MVUniverseBestPractices_V2