Virtual Computing Environment (VCE) Software Guide (M348-23J)

The Open University

Sep 01, 2023

CONTENTS

VCE Cribsheet

- Module code: `M348`
- Presentation code: `23J`
- Jupyter notebook server password / access token: `M348-23J`
- Home directory path inside VCE container: `/home/ou/M348-23J`

Local VCE Settings

- Recommended shared directory name on your computer: `M348VCE`
- Recommended location of shared directory:
  - (Windows): `C:/Users/your_username/Documents/M348VCE`
  - (Mac): `/Users/your_username/Documents/M348VCE` or `~/Documents/M348VCE`
- Browser URL for local VCE (using recommended port):
  - `http://localhost:8348` or `http://127.0.0.1:8348`

Docker command line quick start for running local VCE:

Using a terminal, change directory (`cd`) into the directory you want to share, then create and run a container using a command of the form:

```
docker run -d --name NAME -p PORTS -v VOLUMES IMAGE
```

substituting in the following values:

- **NAME**: `m348vce`
- **PORTS**: `8348:8888`
- **VOLUMES**: `"$(pwd):/home/ou/M348-23J"` (You must retain the quotation marks if there are any spaces in directory name paths. You can also pass in an explicit path rather than the "present working directory" (`$(pwd)`))
- **IMAGE**: `ousefulcoursecontainer/ou-m348:23j`

Alternatively, use Docker Desktop to launch a container from the required image using the above settings; select your `M348VCE` shared folder to be mounted against the `/home/ou/M348-23J` directory.

CHAPTER

ONE

INTRODUCTION

In this module, you will use a module specific virtualised computing environment (VCE) for some of the module activities. We offer two ways of accessing the VCE: as an online hosted version using the Open University's Compute Home service, and as a locally run environment that you can install on your own computer.

The VCE provides a customised computing environment appropriate for your module and an interactive browser based user interface to access the software running inside it. The VCE is implemented using the widely supported Docker container approach.

The hosted online environment is straightforward to use, and is suitable for most students (*Compute Home online VCE*). If you want to use the VCE when you do not have a connection to the Internet, or if you want to gain experience of installing and running a virtualised computing environment using Docker containers, then you should choose to download and install the local environment (*Local VCE quick start*). Further advice on choosing between the two environments is provided below.

Your work in the VCE will be saved within the environment: either in the cloud or on your local machine. If you are using the cloud environment, you should regularly download a copy of your work files, particularly any TMA and EMA related work, to your local machine. For the local VCE, you might also want to back up important files to an alternative location.

This *M348 Virtual Computing Environment Software Guide* includes the following information:

- choosing between the hosted or local VCE
- accessing the hosted VCE
- installing the VCE on your own computer, including testing your installation
- keeping a backup of your work
- accessibility information
- information on finding additional support, and guidance on where to look for or ask for help
- troubleshooting

1.1 Browser requirements

You will access the software tools provided by the virtual computing environment (either the hosted VCE or the locally run VCE) via a web browser. For VCEs running Jupyter environments, the notebooks have been tested extensively with the Chrome web browser. Recent versions of the Firefox, Edge and Safari web browsers should also work; Internet Explorer and older, non-Chromium versions of Edge are *not* supported and notebooks may not work correctly if you use them.

To make it easier to access a locally running VCE, we suggest that you add a browser bookmark for the default web page published by the locally running VCE container. For the remotely hosted VCE, we suggest that you access Compute Home via a bookmark for the module Resources *page on the VLE.*

1.2 Choosing between the OU hosted and the locally run VCE

The online hosted Compute Home service provides the simplest and easiest way of accessing a module's VCE. All you need to access hosted service is an internet connection and a computer running a modern web browser. For the duration of the module, all your work will be stored online in a personal file storage area. However, at the end of the module, you will lose access to the hosted VCE. For many students, Compute Home provides the most convenient way of accessing the M348 VCE.

If you need to access the VCE in an offline environment, or if you prefer not to use the hosted environment, you can run the VCE locally on your own computer. The local environment also provides a way of using the VCE when the module has finished and the online VCE is no longer available.

The online hosted VCE and the locally run VCE provide the same working environment, so your decision as to which environment to use may depend on convenience as much as anything. You may want to make use of both environments, accessing each of them at different times or in different circumstances. However, when doing so, you will have to manage how you synchronise your files across the environments yourself.

1.3 Backing up your work

It is generally regarded as good practice to make backup copies of any files that you would not like to lose. This applies to the contents of the shared folder when using the local VCE, as well as the folders within the hosted VCE.

In the local VCE, only the files you save inside the VCE directory that the shared folder has been mounted against (recommended as `/home/ou/M348-23J`) will be saved to the shared folder on your desktop. All files inside the VCE will persist inside the container until the container is deleted or destroyed.

Backups are regularly made of your files on the hosted VCE. For the local VCE, files will persist in the directory on your host computer that is mounted into the VCE, even if the VCE container is destroyed.

However, it is still good practice, and beneficial to your own peace of mind if nothing else, particularly when working on TMAs or the EMA, for you to back up or grab an archival copy of your files every so often.

1.4 Updates and upgrades

In putting together the VCE, we have tried to ensure that all the software packages and their interconnections run smoothly. Some of the configuration is software version specific, so you are strongly encouraged not to update or upgrade any of the software packages installed within the environment unless instructed to do so by the module team. Software updates and upgrades occasionally introduce changes that result in undesirable software behaviour, sometimes known as 'breaking changes', that cannot always be predicted in advance.

Information regarding any critical updates or changes recommended by the module team will be distributed via the module forums.

1.5 Handling errors — don't panic

Hopefully, you should not see any error messages when installing or running the software provided by the VCE. On completion of each step everything should be working.

You should try to make sure you have the virtual computing environment up and running as soon as you are advised to access it in the module materials or study calendar. If there are any problems then there should be plenty of time to solve them.

If something doesn't appear to be working, try to read through the error messages and see if you can tell what didn't load properly. Most importantly of all, **don't panic**.

Please remember when raising software issues that posting error logs can really help others to try to solve the problem. Saying 'My software is broken/doesn't work/prints out scary red or purple messages' is not helpful. However, sharing those scary messages probably is. Most importantly of all, **don't be embarrassed** about sharing error messages: they often contain the key to the solution of whatever problem caused them.

If you do encounter a problem, the section *Additional support* describes a general strategy for working through the problem and how to ask for help. A later section, *Troubleshooting*, provides more specific guidance for working through particular sorts of issues with the different software applications.

1.6 Where next?

The next section, Section 2, *Compute Home online VCE*, describes how to get started with the Open University online hosted VCE. There are then two sections — Section 3 (*Local VCE quick start*) and Section 4 (*Local VCE detailed guidance*) — that describe how to install a locally run version of the VCE on your own computer. If you want to retain access to the VCE at the end of the module, the local VCE will remain available on your computer until you decide to delete it. You do not need to read those sections if you only wish to use the hosted Compute Home service.

Several sections then follow that describe how to get started using the environment (Section 5), how to make it more accessible to your needs (Section 6), common Jupyter notebook 'gotchas' (Section 7), obtaining additional support if desired (Section 8) and how to troubleshoot any issues that arise (Section 9).

CHAPTER

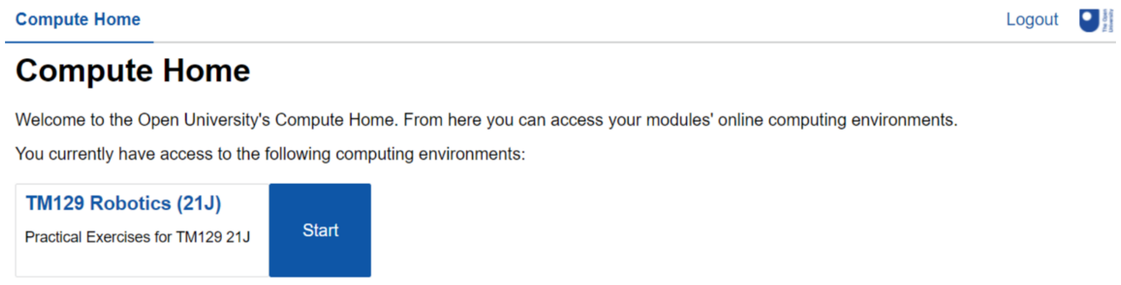
TWO

COMPUTE HOME ONLINE VCE

The Compute Home online hosted VCE is provides the simplest way of accessing a module's VCE. No local software installation is required other than a modern web browser, although you will need an internet connection to access the service. In principle, you should be able to access the online VCE using a smartphone or tablet device as well as a laptop or desktop computer. However, you may find the screen size and on-screen keyboard a limiting factor on usability when using mobile devices. We recommend the use of the Chrome web browser for using the Compute Home online VCE.

A link to the online Compute Home service will be typically be found on the module website in the *Resources* section, often under the heading *Computing Resources*. For modules where the VCE is only required in a particular study block, the *Computing Resources* may appear in a particular block resources section. Click the *Compute Home* link to get started. You should see an entry for your particular module. For example, Figure 2.1 shows a VCE launcher link for the October 2021 presentation of the TM129 module.

Click the *Start* button for the *Applied Statistical Modelling M348 23J* VCE to launch the environment.



Compute Home

Welcome to the Open University's Compute Home. From here you can access your modules' online computing environments.

You currently have access to the following computing environments:

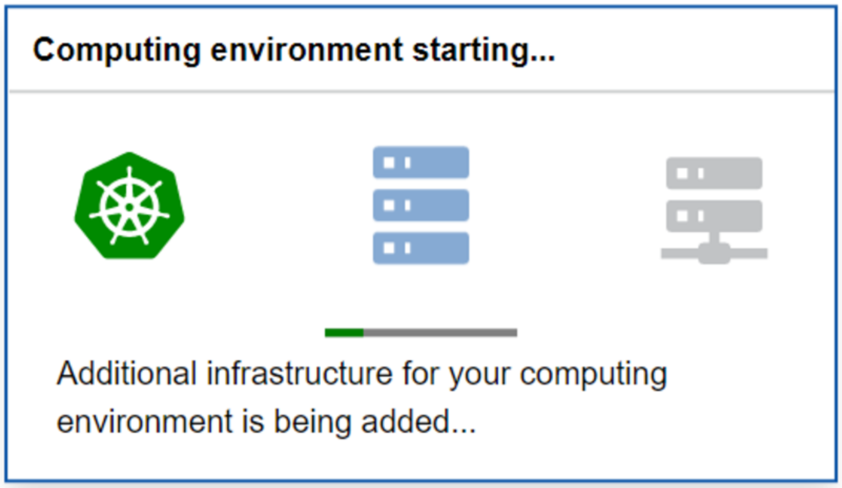
TM129 Robotics (21J)

Practical Exercises for TM129 21J

Figure 2.1: Your Compute Home page. The presentation code will reflect the presentation you are enrolled on

A screenshot of the Compute Home user interface. A single computing environment is shown as being available, identified by the module code and name, in this example *TM129 Robotics*, and presentation code (in the example, 21J, corresponding to the October 2021 presentation).

You may have to wait for several minutes while the environment is set up for you, Figure 2.2. On subsequent visits, access will be quicker.



Computing environment starting...

Additional infrastructure for your computing environment is being added...

Figure 2.2: Waiting while the VCE is set up
A screenshot showing an indicator panel that is displayed when a hosted VCE container is being deployed. The panel display gives an indication of progress across the various stages of deployment.

Once the VCE has loaded, you will be presented with the VCE homepage as described in the section *Using the VCE*.

CHAPTER

THREE

LOCAL VCE QUICK START

Many Open University modules run over several years. In order to keep the software and applications used in a module VCE current, the module VCE may be updated for each presentation.

When installing or accessing the VCE, you will need to make use of the **module code** (`M348`) and the **module presentation code** (`23J`). The module presentation code is constructed from the year and month of presentation. For example, the October 2023 presentation has the code `23J`, where `23` represents the year, and `J` the month (the 10th month of the year, mapped to the tenth letter of the alphabet). *The VCE cribsheet provides a handy summary of these values.*

In many cases, the following quick start instructions should be enough to get you going. Quick start instructions are provided for using the *Docker Desktop graphical user interface*, or, if you prefer, the *command line*.

3.1 Creating a shared folder

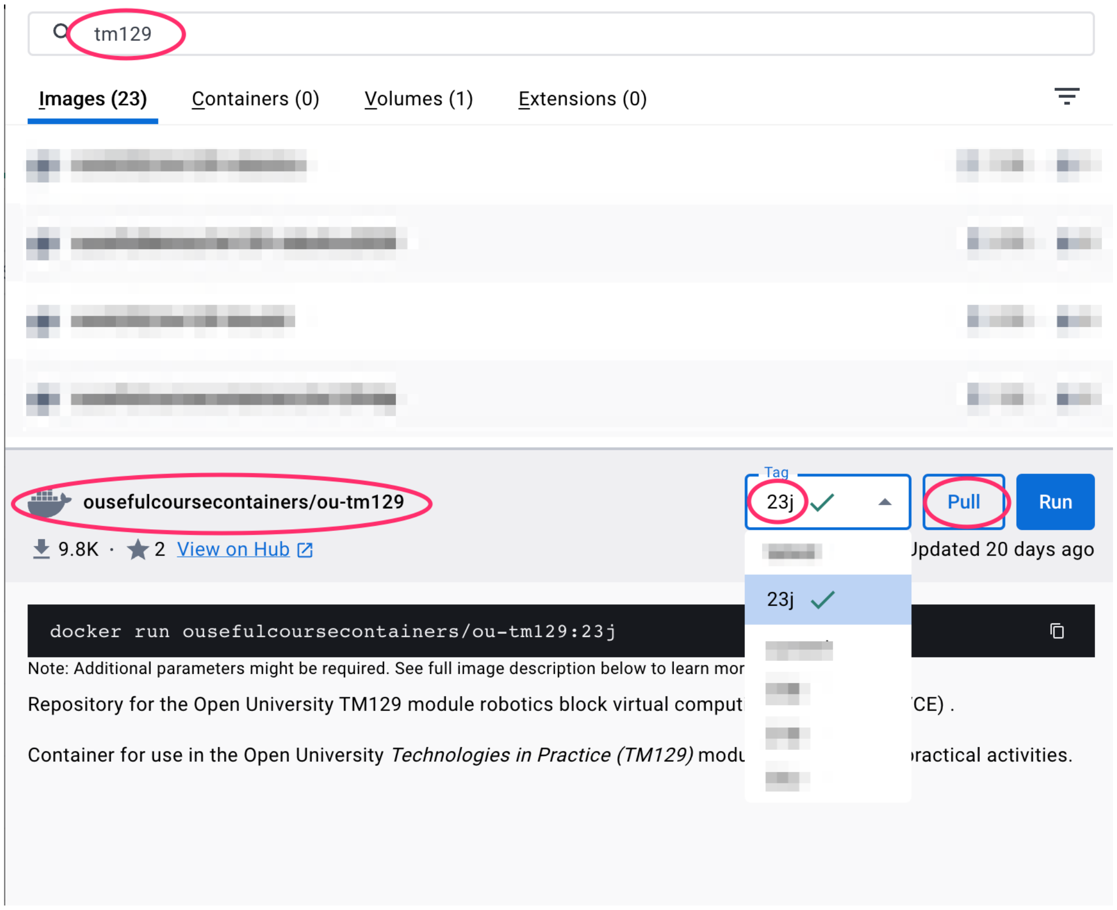
To share files between your desktop computer and the local VCE, we recommend creating a shared directory in your home directory on your host computer with the name `M348VCE`.

We recommend creating the shared folder in your documents folder. On a Windows computer, this might be called something like `C:/Users/your_username/Documents/M348VCE` ; on a Mac, `/Users/your_username/Documents/M348VCE` or `~/Documents/M348VCE`.

3.2 Docker Desktop quick start

- If you do not already have it installed, download and install Docker Desktop from the Docker website
- From the Docker Desktop search bar, Figure 3.1:
  - search for `M348` or `ousefulcoursecontainer/ou-m348:23j`

  - select the image with the tag corresponding to the current presentation code (`23j`) and pull the selected image



```
docker run ousefulcoursecontainers/ou-tm129:23j
```

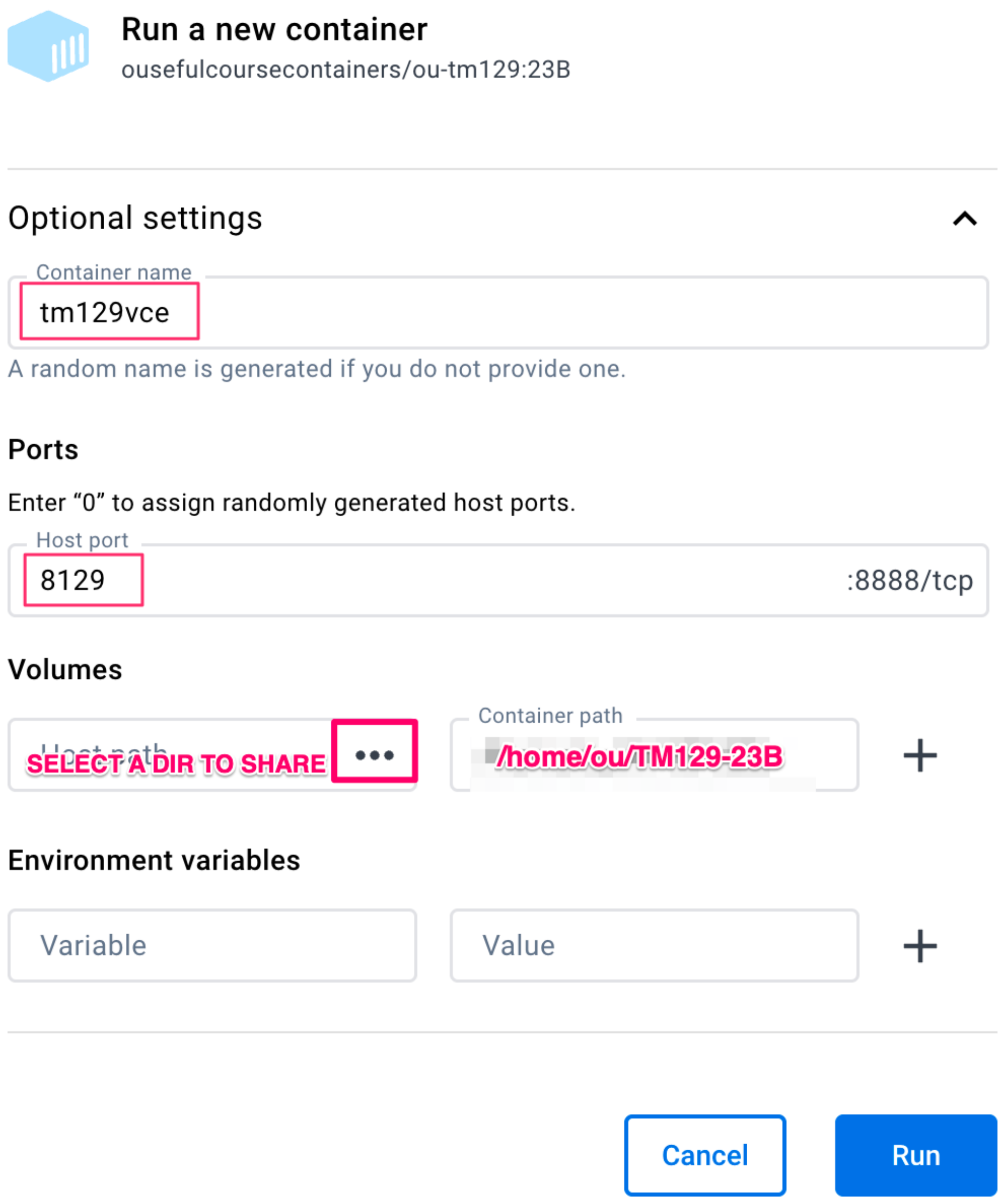
Note: Additional parameters might be required. See full image description below to learn mor

Repository for the Open University TM129 module robotics block virtual comput CE) .

Container for use in the Open University *Technologies in Practice (TM129)* modu ractical activities.

Figure 3.1: Docker image search results in Docker Desktop (as of October 2023, tags will be lower case by convention, e.g. `23j`).
Screenshot showing results of searching for an image in Docker Desktop. A selected image is available in several tagged versions. The `23j` version is shown as selected from a drop down list of available tags.

- From the Docker Desktop images list, select the appropriate container and create a new container with the following optional settings, Figure 3.2:
  - *container name:* `m348vce`
  - *ports:* use host port `8348:8888` as the port mapped from the Jupyter notebook port `:8888/tcp` inside the container
  - *volumes:* select a folder you want to share into the container from your host computer; we recommend creating a shared folder called `M348VCE` on your computer such as `C:/Users/your_username/Documents/M348VCE` (Windows) or `/Users/your_username/Documents/M348VCE` (Mac/Linux). This folder should be mounted against the path `/home/ou/M348-23J` inside the container.



Run a new container
ousefulcoursecontainers/ou-tm129:23B

Optional settings

Container name: tm129vce

A random name is generated if you do not provide one.

Ports

Enter "0" to assign randomly generated host ports.

Host port: 8129 :8888/tcp

Volumes

Host path: SELECT A DIR TO SHARE

Container path: /home/ou/TM129-23B

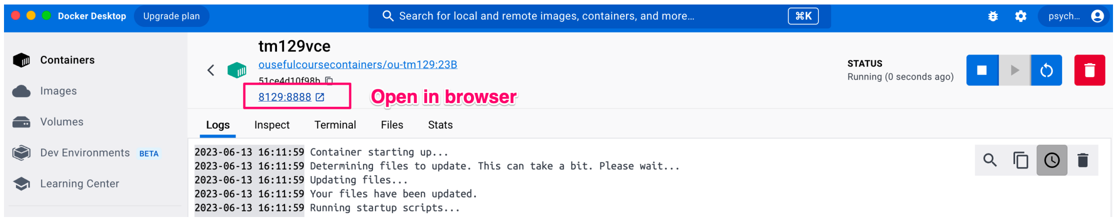
Environment variables

Variable

Value

Figure 3.2: Docker Desktop new container optional settings dialog
Screenshot of the Docker Desktop form for configuring a new container with optional settings. The container name is suggested to be `tm129vce`; the port mapping for `:8888/tcp` is suggested as `8129`; the target for a volume mounted into the container is identified as the uppercase case `/home/ou/TM129-23J` as appropriate for the October 2023 presentation image.

- From the running container page, Figure 3.3, click on the link to the mapped port (using the above defaults, this should point to `http://localhost:8348`.
    - Use the password / access token `M348-23J` (all upper case) to access the notebooks.



```
2023-06-13 16:11:59 Container starting up...
2023-06-13 16:11:59 Determining files to update. This can take a bit. Please wait...
2023-06-13 16:11:59 Updating files...
2023-06-13 16:11:59 Your files have been updated.
2023-06-13 16:11:59 Running startup scripts...
```

Figure 3.3: Docker Desktop running container page
Screenshot of the Docker Desktop panel for a running container. The link to a mapped port is highlighted. Clicking the link will open a browser onto the mapped network location.

- Test the installation by running through any `READ_ME_FIRST.ipynb` style notebooks in the VCE `content/` folder.

3.3 Docker command line quick start

1. Download and install Docker Desktop from the Docker website
2. from a terminal, download the appropriately tagged Docker image using the command:

```
docker pull ousefulcoursecontainer/ou-m348:23j
```

3. from a terminal, create a working directory you want to share with container (for example, at `C:/Users/your_username/Documents/M348VCE` (Windows) or `/Users/your_username/Documents/M348VCE` (Mac/Linux)); change directory (`cd`) into the directory you want to share, then create and run a container using a command of the form:

```
docker run -d --name NAME -p PORTS -v VOLUMES IMAGE
```

and substitute in the following values:

- **NAME**: `m348vce`
- **PORTS**: `8348:8888`
- **VOLUMES**: `"$(pwd):/home/ou/M348-23J"` (You must retain the quotation marks if there are any spaces in directory name paths. You can also pass in an explicit path rather than the "present working directory" (`$(pwd)`))
- **IMAGE**: `ousefulcoursecontainer/ou-m348:23j`

4. In a browser, navigate to `http://localhost:8348`
5. Use the password / access token `M348-23J` to enter the notebook server.

6. Test the installation by running through any `READ_ME_FIRST.ipynb` style notebooks in the VCE content/ folder.

CHAPTER

FOUR

LOCAL VCE DETAILED GUIDANCE

This section provides more detailed guidance on installing and troubleshooting the VCE installation.

4.1 Hardware requirements

The virtual computing environment can be installed onto any computer that can run the Docker application, such as a desktop operating system (Microsoft Windows, macOS, Linux) but not tablet computers or most Chromebooks. To run the virtual computing environment, you will need a 64-bit computer processor and at least 4 GB of memory. You will also need at least 20 GB of free disk space to store the virtual computing environment and the data files you will be working with using this VCE.

4.2 Creating your shared folder

The VCE runs as an isolated virtual machine on your computer. You can share files between the VCE and your host operating system by means of a shared folder. For the shared folder, we recommend creating a folder on your computer called `M348VCE` in your documents folder. On a Windows computer, this might be called something like `C:/Users/your_username/Documents/M348VCE`; on a Mac, `/Users/your_username/Documents/M348VCE` or `~/Documents/M348VCE`.

4.3 Installing the local VCE

These instructions describe how to:

- download and install the Docker application if it is not already installed and available
- download the module specific Docker container image that contains all the software applications needed for the module

To familiarise yourself with the installation sequence, we recommend that you read through the installation process for your host operating system and the description of testing your installation at least once before carrying out the process on your own computer.

4.3.1 Microsoft Windows

For Windows 10 or 11 Home or Pro, Enterprise or Education, all at version 21H2 or higher, follow the WSL2 backend instructions on the Docker website for how to Install Docker Desktop on Windows (you may need to scroll down to find the detailed instructions). This will require you to:

- download the Docker installation package
- double-click the installation package to run it
- follow the Windows Subsystem for Linux (WSL2) instructions to make sure that WSL2 is installed and enabled (this may require restarting your computer)
  - for older Windows systems, you may need to follow this guidance on how to install WSL2
- check your virtualisation settings:
  - note: the 'Turn Windows features on or off' dialog box can be obtained by doing the following: Click on Start and start typing 'Turn Windows features on or off' into the search bar;
  - ensure the following Windows settings are enabled: *Virtual Machine Platform* and *Windows Subsystem for Linux*
- start the Docker application:
  - search for *Docker*, and select *Docker Desktop* in the search results.

Windows: accessing a terminal command-line interface

To run the `wsl --install` command and Docker commands on the command line, you will need access to PowerShell.

In Windows, click on Start and start typing PowerShell into the search bar. Open the PowerShell application that should be listed as a result.

4.3.2 Apple macOS

Apple Mac users should follow instructions on the Docker website for how to Get started with Docker Desktop on Mac (you may need to scroll down for detailed instructions).

To find out whether your Mac uses an Intel or Apple silicon chip:

- click the Apple menu and select *About this Mac*
  - check the chip type (XXX Intel YYY or Apple M1 (or above))
  - for older Macs, the chip is described in the Overview tab of the *About this mac* dialogue

This will require you to:

- download the Docker installation package
- double-click the installation package to run it (we recommend using the default recommended settings)
- start the Docker application.

You should now be able to download the VCE container image from the Docker Hub.

To make it easier to access the Docker Desktop, open the *Applications* folder, find the *Docker* application and drag the application onto the Dock, You should now be able to launch the Docker Desktop application from the Dock.

macOS: accessing a terminal command-line interface

To run Docker commands on the command line, you will need access to a terminal.

Open the *Applications* folder and find the *Utilities* folder inside it. The *Terminal* application should be in that (*Applications/Utilities*) folder. Drag the *Terminal* onto the Dock so you can access it more easily in future.

4.3.3 Linux

Linux users may find that Docker is already installed as part of their Linux distribution. If it is not already installed, follow instructions for how to install Docker Desktop on Linux platforms.

To launch the Docker Desktop application, open your *Applications* menu in Gnome/KDE Desktop and search for Docker Desktop.

Linux: accessing a terminal command-line interface

To run Docker commands on the command line, you will need access to a terminal.

In the *Applications* menu, the *Terminal* can usually be found in the *System* or *Accessories* menu area.

4.4 Downloading the VCE Docker image

A Docker image provides a static template for creating an instance of a personal VCE in a running Docker container. The easiest way to download a Docker image is by using the Docker Desktop search toolbar, Figure 4.1.

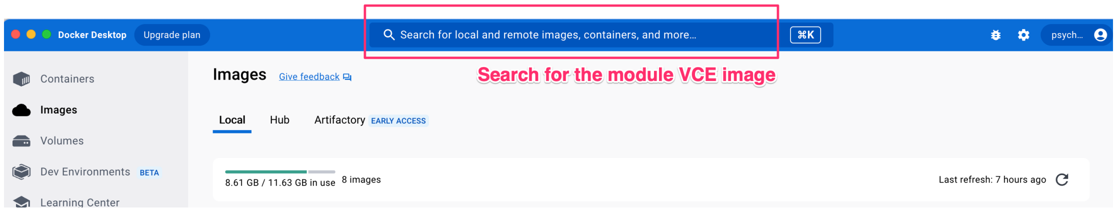


Figure 4.1: The Docker Desktop search toolbar
Screenshot of the Docker Desktop search toolbar.

Search for an image using the module code, as shown in Figure 4.2.

- search for and select the `ousefulcoursecontainer/ou-m348:23j` image from the search results; then from the drop down tag list select the tag that matches the presentation code for this presentation of the module: `23j`. Clicking the *Pull* button will then download the image from the Docker Hub repository to your computer. *The Docker image may take some time to download (up to 20 minutes depending on your network connection).* You should only need to download the image once.

You can also download an image from the command line by running the following command from the command line:

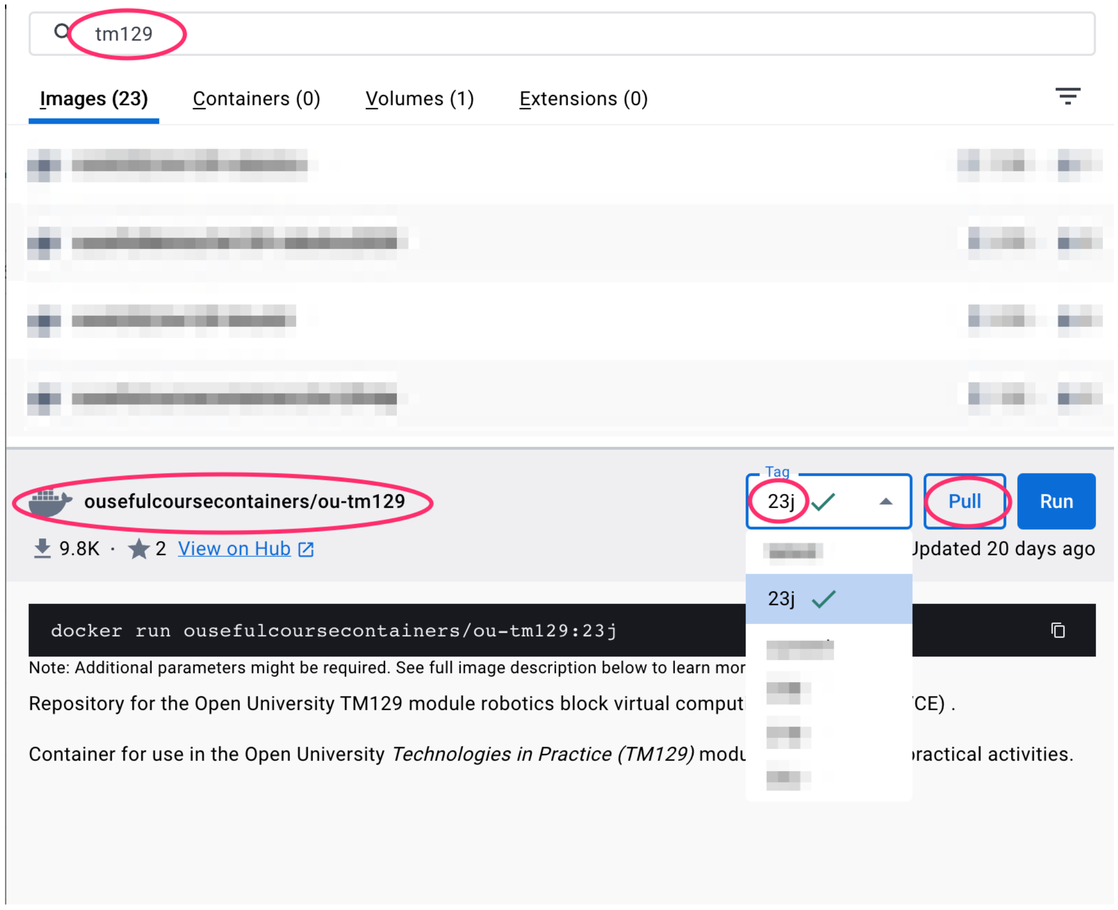
```
docker pull ousefulcoursecontainer/ou-m348
```

Enter the complete command using lower-cased characters.

4.5 Running the Docker container

The VCE is accessed from a running Docker container. A container is a virtual machine instance created from a previously downloaded Docker container image.

The simplest way of running and managing containers is to use the Docker Desktop application.



```
docker run ousefulcoursecontainers/ou-tm129:23j
```

Note: Additional parameters might be required. See full image description below to learn mor

Repository for the Open University TM129 module robotics block virtual comput CE) .

Container for use in the Open University *Technologies in Practice (TM129)* modu ractical activities.

Figure 4.2: height: 4.6875in
Docker image search results in Docker Desktop (as of October 2023, tags will be lower case by convention, e.g. `23j`*).*
Screenshot showing results of searching for an image in Docker Desktop. A selected image is available in several tagged versions. The `23j` version is shown as selected from a drop down list of available tags.

4.5.1 Creating a new container

From the *Images* view in the Docker Desktop, identify the appropriate image and click the play button, Figure 4.3.

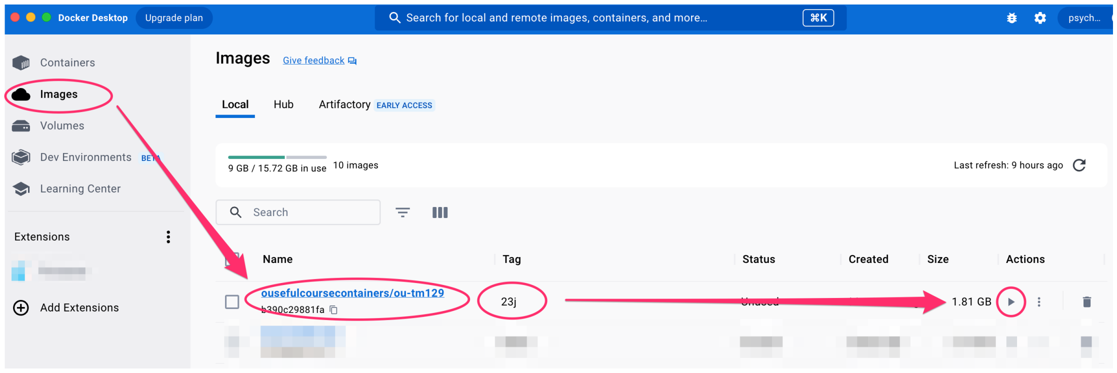


Figure 4.3: Docker Desktop images page
Screenshot of the Docker Desktop images view. The TM129 image is identified and the associate play button for creating a running container from it is indicated.

From the *Run a new container* dialogue, use settings of the following form:

- *container name:* `m348vce` (all lower case)

- *ports:* a mapping from the port number inside the container (for example, against `:8888/tcp` for a conventional Jupyter server) onto a location you can access in your browser; use host port 0 to randomly allocate a port, or a port number derived from the module code such as `8348` (8 followed by the numerical part of the M348 module code)).

- *volumes:* using the file dialogue (click on the three dots) select the folder you want to share into the container from your host computer, such as the folder you created in your home computer documents folder previously. This folder can be shared by mounting it against the path `/home/ou/M348-23J` inside the container (use your module code).

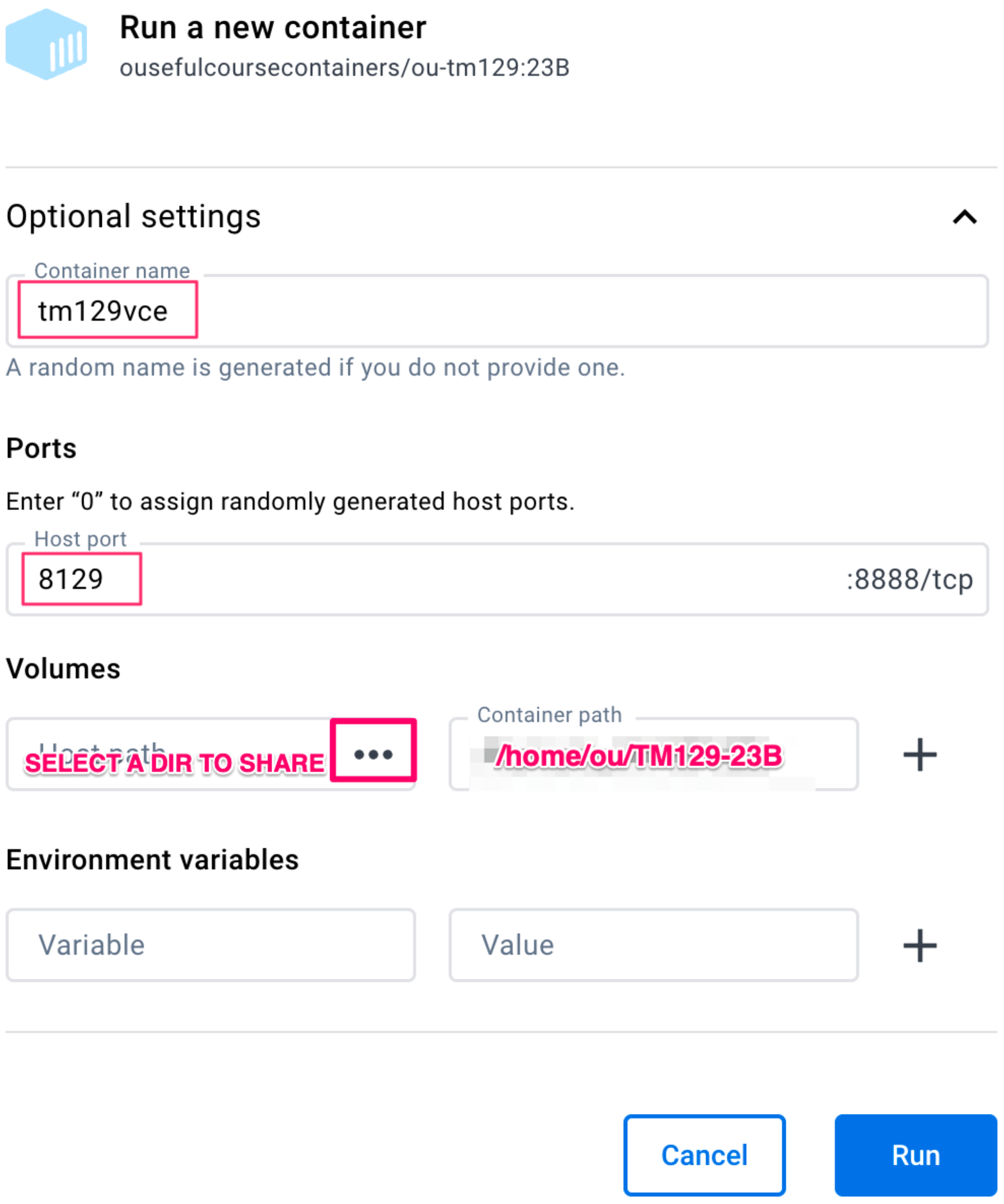
Note: the path inside the container is case sensitive. Use upper case for the module directory inside the container.

Note:

- you cannot have two containers with same name. To reuse a container name, you must first stop and delete the container with the name you want to reuse.
- once a container has been created, you cannot change the port mapping, or mount additional volumes into the container.

You can view the user interface published by the running container, click on the forwarded link that appears in the container status area at the top of the container information page, Figure 4.5.

The first time you try to access the notebook user interface, you will be presented with a Jupyter server login screen. The password / access token is `M348-23J` (all upper case).



Run a new container

ousefulcoursecontainers/ou-tm129:23B

Optional settings

Container name: tm129vce

A random name is generated if you do not provide one.

Ports

Enter "0" to assign randomly generated host ports.

Host port: 8129 :8888/tcp

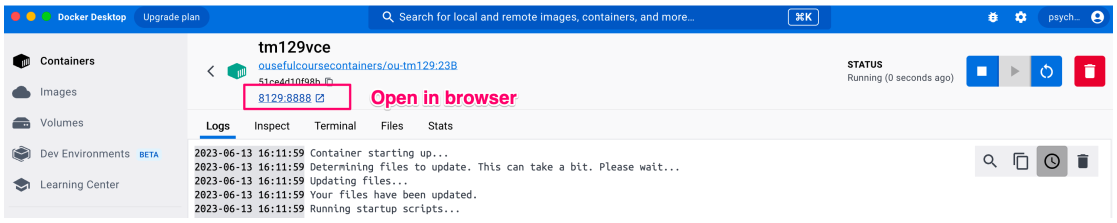
Volumes

Host path: SELECT A DIR TO SHARE

Container path: /home/ou/TM129-23B

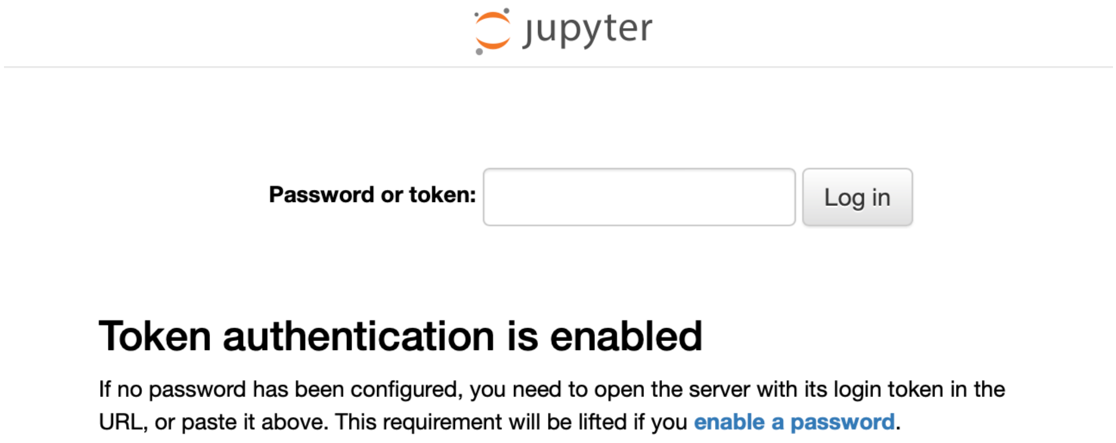
Environment variables

Figure 4.4: Docker Desktop new container optional settings dialog
Screenshot of the Docker Desktop form for configuring a new container with optional settings. The container name is suggested to be `tm129vce`; the port mapping for `:8888/tcp` is suggested as `8129`; the target for a volume mounted into the container is identified as the uppercase case `/home/ou/TM129-23J` as appropriate for the October 2023 presentation image.



```
2023-06-13 16:11:59 Container starting up...
2023-06-13 16:11:59 Determining files to update. This can take a bit. Please wait...
2023-06-13 16:11:59 Updating files...
2023-06-13 16:11:59 Your files have been updated.
2023-06-13 16:11:59 Running startup scripts...
```

Figure 4.5: Docker Desktop running container page
Screenshot of the Docker Desktop panel for a running container. The link to a mapped port is highlighted. Clicking the link will open a browser onto the mapped network location.



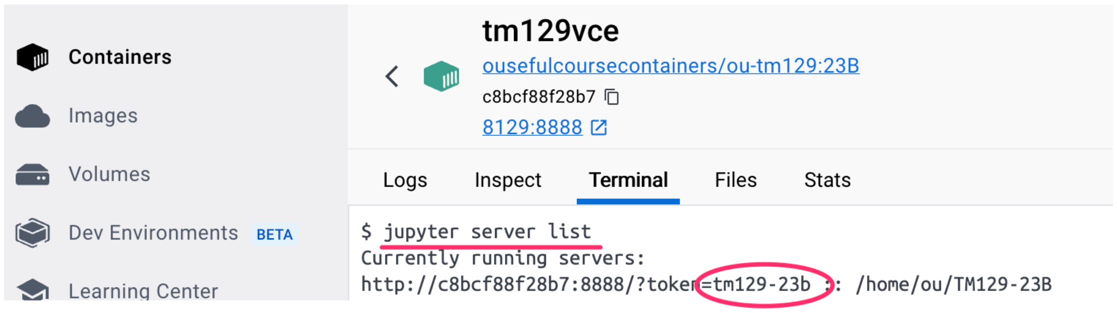
Token authentication is enabled

If no password has been configured, you need to open the server with its login token in the URL, or paste it above. This requirement will be lifted if you enable a password.

Figure 4.6: The Jupyter server password token prompt when the notebook server is first accessed
Screenshot of the Jupyter notebook server password / prompt page when the notebook server is first accessed. No suggested password / prompt is indicated.

Note: you can find the password token by running the following command on the command line *inside* the container, Figure 4.7 (see also *Opening a terminal inside a running container*):

```
jupyter server list
```



```
$ jupyter server list
Currently running servers:
http://c8bcf88f28b7:8888/?token=tm129-23b :: /home/ou/TM129-23B
```

Figure 4.7: Finding the token to use with a running Jupyter notebook server

As well as using Docker Desktop to create and manage containers, you can also use the command line. See *Docker command line quick start*.

Make sure that you use the correct module code and presentation tag and the correct case when specifying the image from which the container instance is created.

You should be able to access the VCE user interface in your browser at the URL `http://localhost:8348`.

See *Managing Docker from the terminal command line* for more examples of controlling Docker from the command line.

4.5.2 Starting, stopping and restarting a container

Containers can be managed from the Containers area of the Docker Desktop, Figure 4.8.

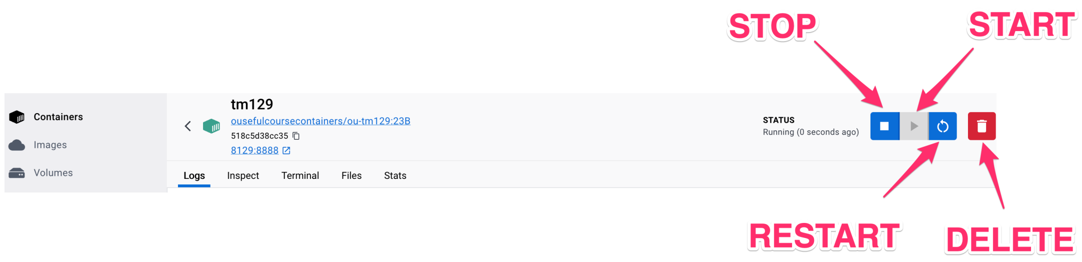
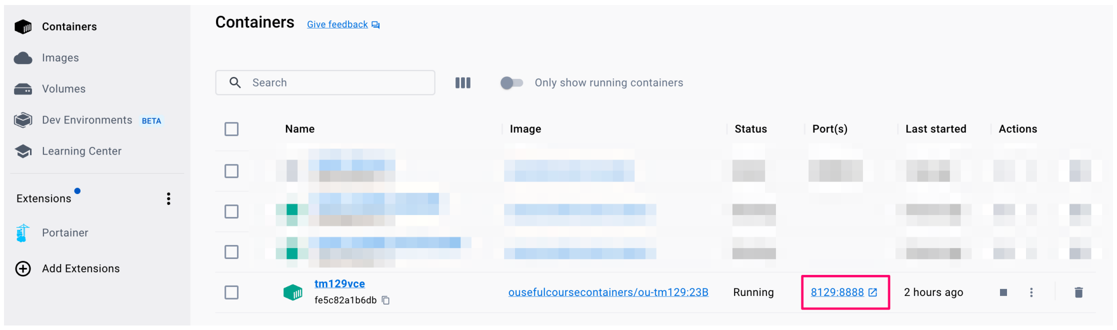


Figure 4.8: Docker Desktop running container page, highlighting the container controls
Screenshot of Docker Desktop running container view, Several control buttons are highlighted: the Stop button, the Start button, the Restart button and the Delete button.

Containers may also be stopped, started and deleted from the *Containers* listing, Figure 4.9.

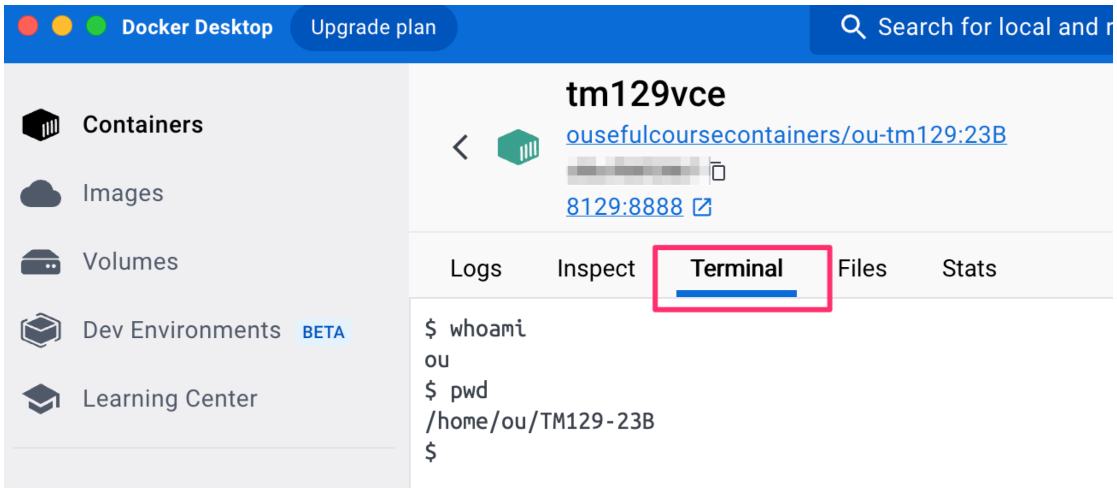


Name	Image	Status	Port(s)	Last started	Actions
tm129vce fe5c82a1b6db	ousefulcoursecontainers/ou-tm129:23B	Running	8129:8888	2 hours ago	

Figure 4.9: Docker Desktop container listing page
Screenshot of the Docker Desktop containers page. All but one container listings are blurred out, leaving the TM129 container as the only readable item. Within this item, a mapped ports link is highlighted that indicates the container port 8888 is mapped to localhost port 8129.

4.5.3 Opening a terminal inside a running container

You can open a terminal inside a running container from a running container tab inside Docker Desktop, Figure 4.10.



```
$ whoami
ou
$ pwd
/home/ou/TM129-23B
$
```

Figure 4.10: Opening a terminal into a container using Docker Desktop

It is also possible to access the command-line *inside* a running container from the command-line on your host computer by running a command of the form.

```
docker exec -it container_name /bin/bash
```

For example, to connect to a running container with the container name `m348vce`, use the command:

```
docker exec -it m348vce /bin/bash
```

By default, you will be logged in to the container as the default container user. You can log in to the container as the root user by using the `-u/--user 0` (user ID `0`) or `-u/--user root` flag:

```
docker exec -it --user root m348vce /bin/bash
```

4.5.4 Managing Docker from the terminal command line

As well as managing your Docker images and containers from the Docker Desktop graphical user interface, you can also manage Docker images and containers from the terminal command line interface.

The core commands are as follows, although they may be modified using various flags:

Action	Terminal command
Pull container image	`docker pull IMAGENAME`
Create a container from an image using a specified container name	`docker run --name CONTAINER_NAME IMAGENAME`
Stop container	`docker stop CONTAINER_NAME`
Start a previously stopped container	`docker start CONTAINER_NAME`
Restart a container	`docker restart CONTAINER_NAME`
Delete a container	`docker rm CONTAINER_NAME`
Show running containers	`docker ps`
Show specific running container	`docker ps --filter "name=CONTAINER_NAME"`
Show ports exposed by a container	`docker port CONTAINER_NAME`
Run a command inside a container	`docker exec -it CONTAINER_NAME COMMAND`
Access the command line inside a container	`docker exec -it CONTAINER_NAME /bin/bash`
List all containers	`docker container --ls --all`

The following flags may also be used with the `docker run` command:

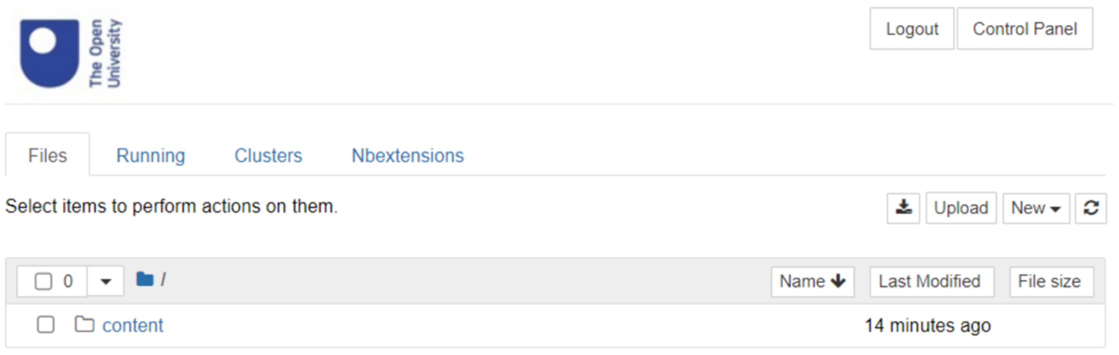
Action	Flags
Create container name	`--name CONTAINER_NAME`
Specify volume mount points	`-v/--volume PATH/ON/HOST:/PATH/IN/CONTAINER`
Port mapping	`-p/--publish HOSPORT:CONTAINER_PORT`
Automatically remove container when it exits	`--rm`
Run container in background	`-d/--detach`
Set environment variable inside container	`-e/--env ENV_VARIABLE="ENV VALUE"`

CHAPTER

FIVE

USING THE VCE

When the VCE is fully initialised, you will see the contents page for the Jupyter notebook environment as shown in Figure 5.1.



Select items to perform actions on them.

14 minutes ago

Figure 5.1: The Jupyter notebook contents page
A screenshot showing a classic Jupyter notebook homepage with a single content directory displayed.

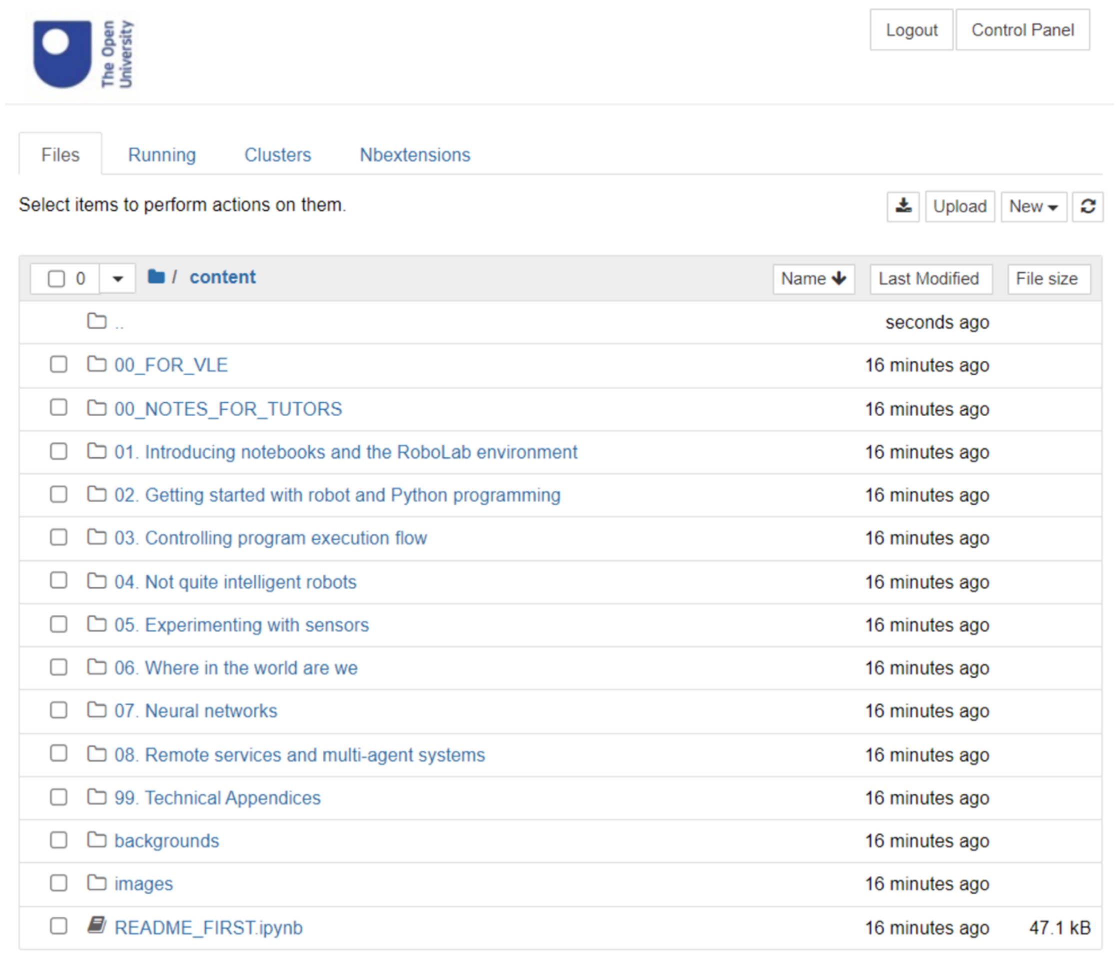
The exact contents of this page depends on the module. For example, Figure 5.2) shows the initial contents page for the module TM129. Click on a folder to see the file contents within it. Notebooks are self-contained and should be worked through in order. You will be directed to appropriate notebooks from the study planner or module materials as you work your way through the module.

5.1 First steps when using the VCE

Many VCEs will include a `READ_ME_FIRST.ipynb` notebook in the `content/` folder that can be used test the VCE installation. You are advised to work through any such notebook.

To leave the Compute Home hosted VCE, click the *Logout* link towards the top right-hand corner of the page. This will shut down the VCE, saving your work before it does so. You will also be automatically logged out of the hosted VCE after a period of inactivity.

When you return to the VCE and reopen a notebook, you will find your work saved from your previous sessions, although you may find you that a new kernel session has been started (see *Jupyter notebook*



Select items to perform actions on them.

/ content

Name	Last Modified	File size
..	seconds ago	
00_FOR_VLE	16 minutes ago	
00_NOTES_FOR_TUTORS	16 minutes ago	
01. Introducing notebooks and the RoboLab environment	16 minutes ago	
02. Getting started with robot and Python programming	16 minutes ago	
03. Controlling program execution flow	16 minutes ago	
04. Not quite intelligent robots	16 minutes ago	
05. Experimenting with sensors	16 minutes ago	
06. Where in the world are we	16 minutes ago	
07. Neural networks	16 minutes ago	
08. Remote services and multi-agent systems	16 minutes ago	
99. Technical Appendices	16 minutes ago	
backgrounds	16 minutes ago	
images	16 minutes ago	
README_FIRST.ipynb	16 minutes ago	47.1 kB

Figure 5.2: Notebook directory listing organised by study week
A screenshot of a classic Jupyter notebook home page file listing show directories in the content/ directory path. Multiple directories are listed in alphanumeric order. The directory names indicate the study week number and the general topic of study for that week. For example, “04. Not quite intelligent robots”.

gotchas).

At the end of each unit or block, or whenever you have submitted notebook related assessment material, we **strongly** recommend that you download a copy of the completed Jupyter notebooks. On completion of the module, you may lose access to the hosted VCE and your hosted work files. To keep access to any changes you have made to your notebooks, you will need to download copies of them to your own computer.

5.2 Non-appearance of notebooks in the local VCE

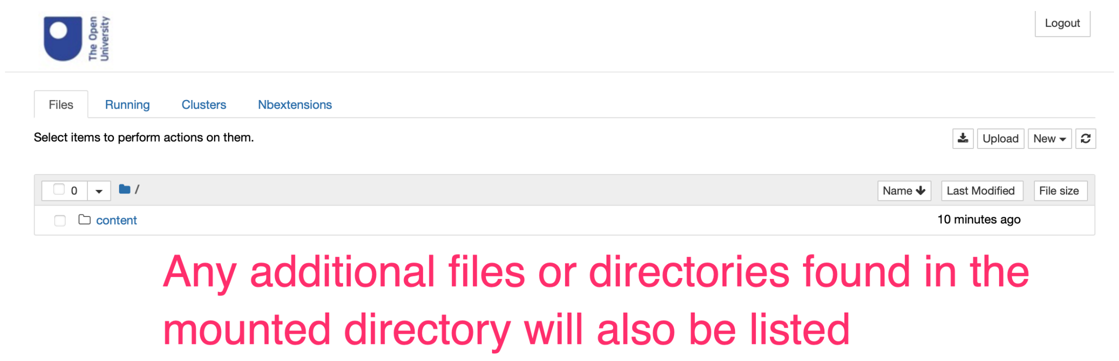
In the local VCE, if the classic Jupyter notebook server application is not available at its default address, there are three possibilities:

1. the application is *not* running
2. the application *is* running, *but* on an incorrectly configured port inside the VCE
3. the application *is* running, and on the correct port, *but* the port or the proxy used to expose it is being blocked in some way (for example, by an overly aggressive anti-virus or firewall package).

See *Troubleshooting* for more guidance on resolving problems or issues with the VCE.

5.3 Using the classic Jupyter notebook environment

The primary user interface for the M348 (23J) VCE is the classic Jupyter notebook home page, Figure 5.3.



Select items to perform actions on them.

Figure 5.3: Classic Jupyter notebook homepage
A screenshot of the Jupyter notebook home page showing a single directory called "content". A label on the screenshot reads: "Any additional files or directories found in the mounted directory will also be listed".

You will typically access the VCE related activities through the notebook home page. Any preloaded materials will be available in the `content` folder. Materials that have been made available by other means, such as via the VLE, will need to be uploaded directly to the VCE, or added via the shared folder when running the local VCE.

Notebooks will typically be organised by block, unit, study week or topic and may be grouped into corresponding folders.

Click on that notebook to open it. Follow the instructions contained therein to test your VCE installation.

When you open a new Jupyter notebook, it creates a new programming environment kernel (for example, a Python or R code environment). The kernel attached to a notebook executes the code contained within that notebook. Code executed in one notebook is isolated from code executed in every other notebook, unless you have open the same notebook in multiple tabs.

The notebook execution model may lead to unexpected behaviours for the unwary if you have the same notebook open in multiple browser tabs, or if you execute code cells out of cell order. See *Jupyter notebook gotchas* for more details.

Follow the guidance on the study planner or in your study materials for a reminder as to when it makes sense to study each notebook.

5.3.1 Uploading files

To upload a file to the VCE, move to the directory you want to upload the file to, and click on the 'upload' button on the menu in your browser, Figure 5.4.

5.3.2 Downloading files

You can download a single notebook to your computer desktop from either the hosted or the local VCE. To download a notebook your currently have open, use the `File -> Download as -> Notebook` menu option, Figure 5.5. This will download the notebook, including all the outputs.

You can also download multiple notebooks from the hosted VCE, or export notebooks from a container with no shared volume, using the "Download as zip" button on the notebook homepage, Figure 5.6.

5.4 Unzipping compressed archive files

A common way of transporting large files or bundles of files is to compress them into a single compressed archive file, such as a `.zip` file.

To unzip a file called `filename.zip` in the VCE, click on the *New* button from the notebook homepage and select *Terminal* to open a terminal. In the terminal, use the command `unzip filename.zip` and press enter: the file will be unzipped into the current directory. *Depending on the contents of*

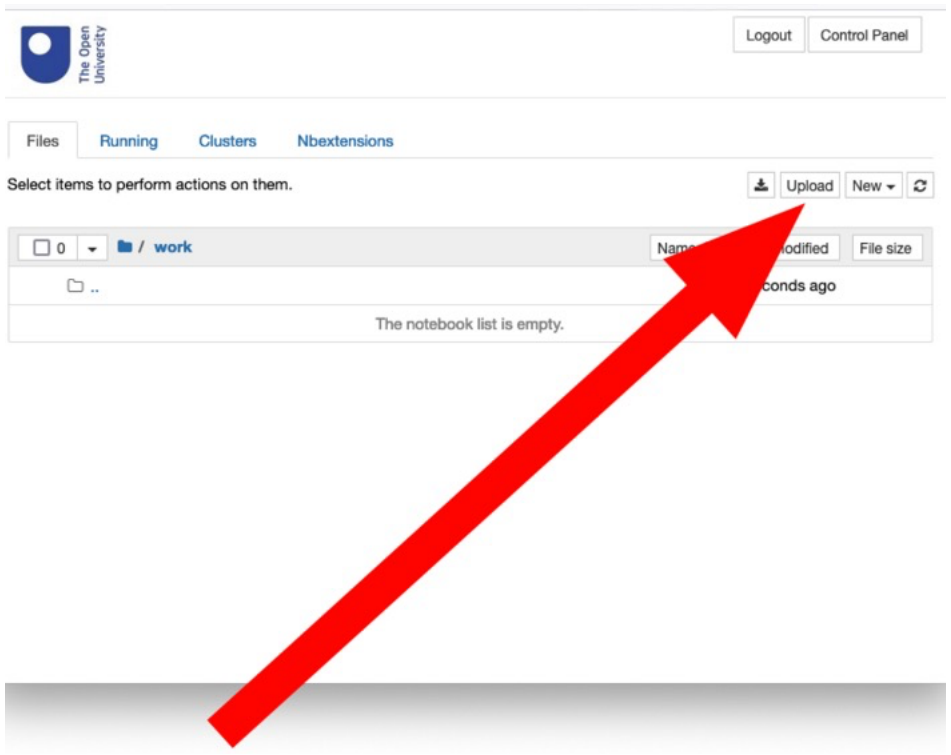
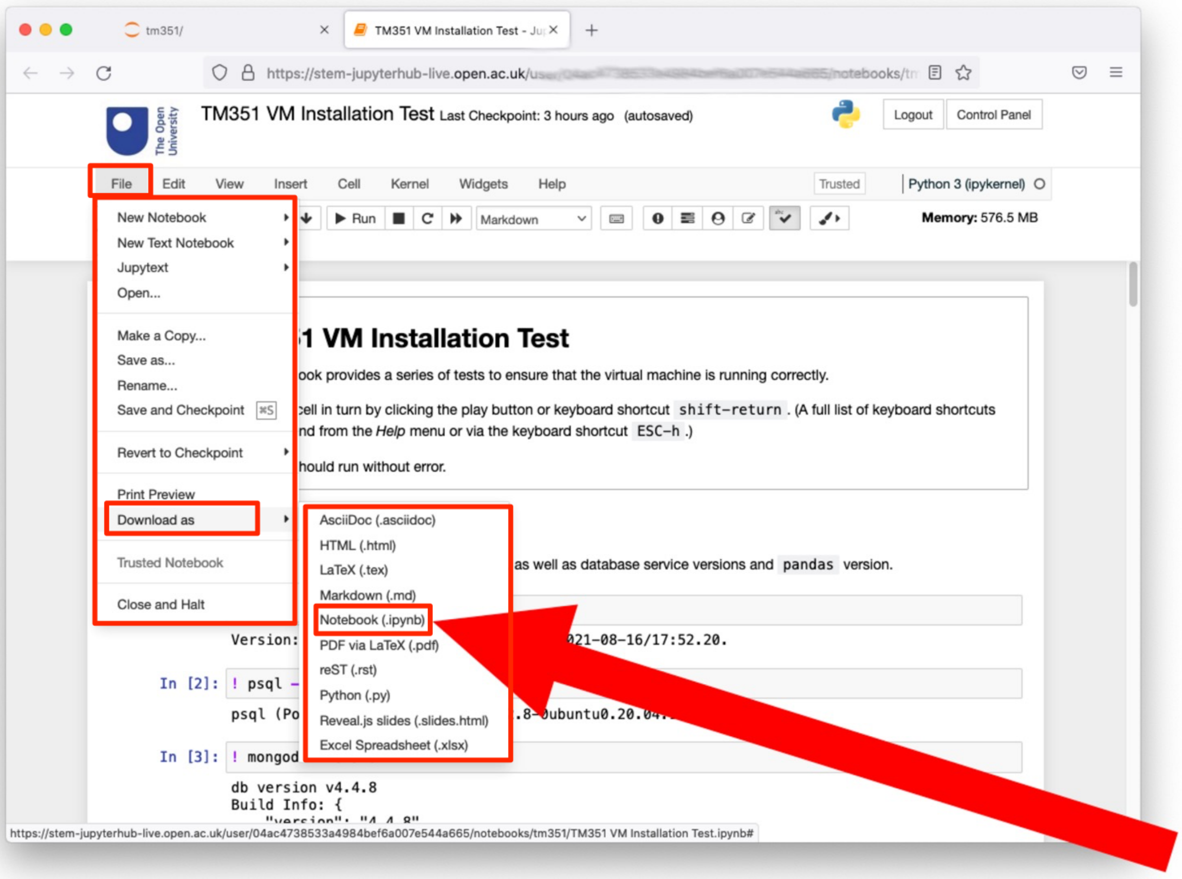


Figure 5.4: Uploading files to a classic Jupyter notebook environment
Screenshot of a classic Jupyter notebook homepage, with the "Upload" button indicated towards the top right hand corner of the display.



1 VM Installation Test

ook provides a series of tests to ensure that the virtual machine is running correctly.

cell in turn by clicking the play button or keyboard shortcut `shift-return`. (A full list of keyboard shortcuts nd from the *Help* menu or via the keyboard shortcut `ESC-h`.)

hould run without error.

as well as database service versions and `pandas` version.

```
Version:                                    021-08-16/17:52.20.

In [2]:  ! psql -
         psql (Po                           .8-0ubuntu0.20.04.

In [3]:  ! mongod

         db version v4.4.8
         Build Info: {
```

Figure 5.5: Classic Jupyter notebook homepage, with download archive button highlighted
Screenshot of a classic notebook , with the 'File menu' open, the "Download as" option selected, and the "Notebook (.ipynb)" submenu option indicated.

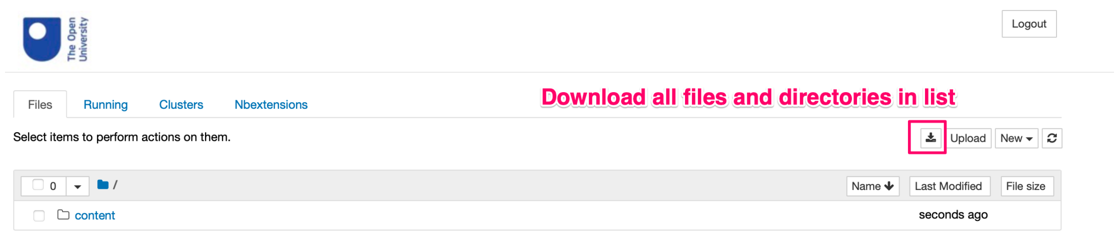


Figure 5.6: Classic Jupyter notebook homepage, with download archive button highlighted
Screenshot of the classic notebook home page, with the "Download as zip" button (a down arrow above a horizontal bar) highlighted.

the zip file, the unzipped files may be contained in a newly created directory within the current directory. Close the terminal.

If the `.zip` file is not in the VCE home directory, use the `cd PATH` command to change directory to the required location before running the `unzip` command, or specify the path to the file (`unzip PATH/TO/filename.zip`).

CHAPTER

SIX

CLASSIC NOTEBOOK ACCESSIBILITY

The notebooks are rendered within a browser as HTML. Instructional text in notebook Markdown cells are directly readable by screen readers. Code cells are contained within HTML group elements and need to be internally navigated to.

Most of the Jupyter notebook features are keyboard accessible. Several optional extensions provide further support in terms of visual styling and limited audio feedback support.

If you struggle to use the VCE for any reason, including but not limited to incompatibility with any tools you may use to improve software access or usability, please raise an issue in the module forums or contact your tutor.

6.1 Keyboard interface

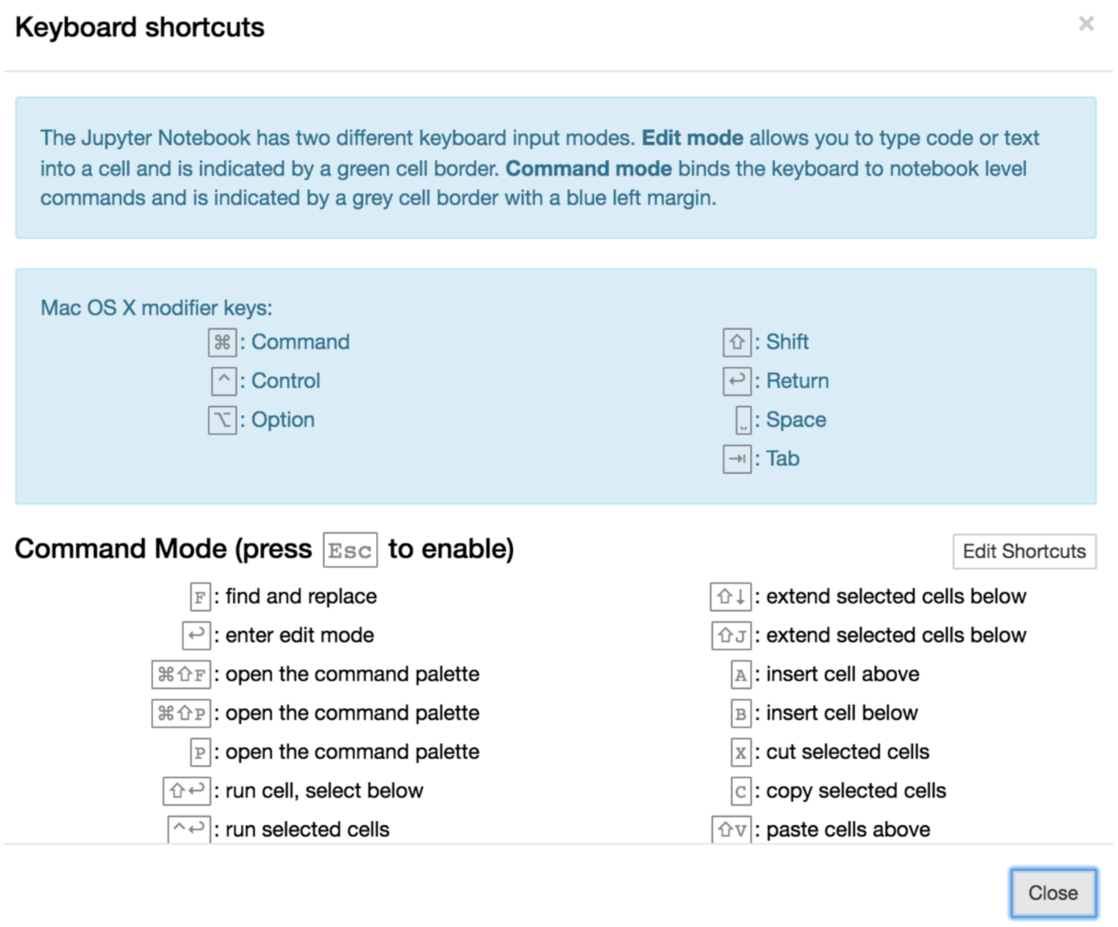
The Jupyter notebook interface supports a wide range of pre-defined keyboard shortcuts to menu and toolbar options. The shortcuts can be displayed using the Keyboard Shortcuts item from the notebook Help menu or via the Esc-H keyboard shortcut Figure 6.1.

You can also add additional shortcuts and/or edit exist shortcuts via the Edit Keyboard Shortcuts menu item as shown in Figure 6.2.

6.2 Magnification

The apparent size of the notebook contents in general can be zoomed using standard browser magnification tools.

Alternatively, use operating systems tools such as Windows Magnify or the macOS Zoom Window, or other assistive software.



Keyboard shortcuts

×

The Jupyter Notebook has two different keyboard input modes. **Edit mode** allows you to type code or text into a cell and is indicated by a green cell border. **Command mode** binds the keyboard to notebook level commands and is indicated by a grey cell border with a blue left margin.

Mac OS X modifier keys:

⌘: Command	⇧: Shift
^: Control	↩: Return
⌥: Option	␣: Space
	⇥: Tab

Command Mode (press Esc to enable)

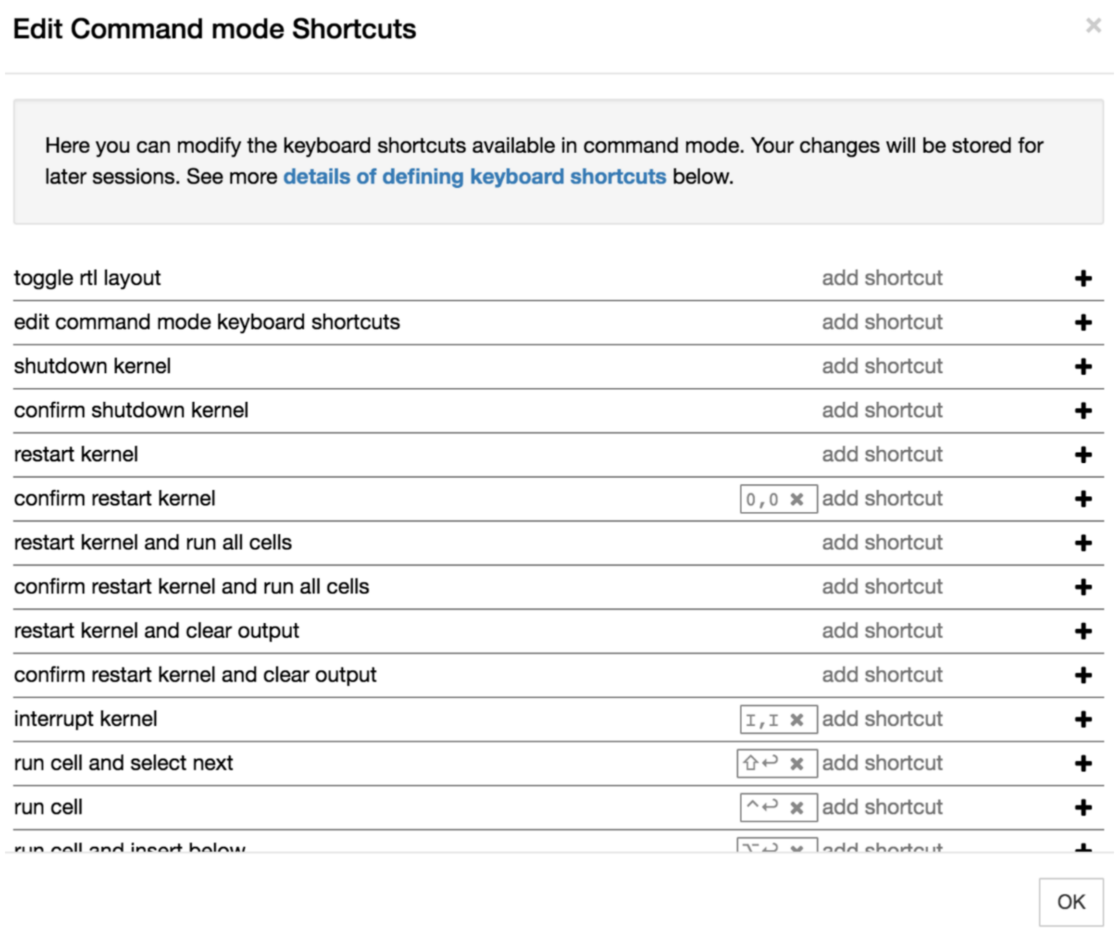
Edit Shortcuts

F: find and replace	⇧↓: extend selected cells below
↩: enter edit mode	⇧J: extend selected cells below
⌘⇧F: open the command palette	A: insert cell above
⌘⇧P: open the command palette	B: insert cell below
P: open the command palette	X: cut selected cells
⇧↩: run cell, select below	C: copy selected cells
^↩: run selected cells	⇧V: paste cells above

Close

Figure 6.1: The Jupyter notebook ‘Keyboard shortcuts’ dialogue box

The Jupyter notebook ‘Keyboard shortcuts’ dialogue box



Edit Command mode Shortcuts

Here you can modify the keyboard shortcuts available in command mode. Your changes will be stored for later sessions. See more **details of defining keyboard shortcuts** below.

toggle rtl layout		add shortcut	+
edit command mode keyboard shortcuts		add shortcut	+
shutdown kernel		add shortcut	+
confirm shutdown kernel		add shortcut	+
restart kernel		add shortcut	+
confirm restart kernel	0,0 ✖	add shortcut	+
restart kernel and run all cells		add shortcut	+
confirm restart kernel and run all cells		add shortcut	+
restart kernel and clear output		add shortcut	+
confirm restart kernel and clear output		add shortcut	+
interrupt kernel	I,I ✖	add shortcut	+
run cell and select next	⇧↩ ✖	add shortcut	+
run cell	^↩ ✖	add shortcut	+
run cell and insert below	⌥↩ ✖	add shortcut	+

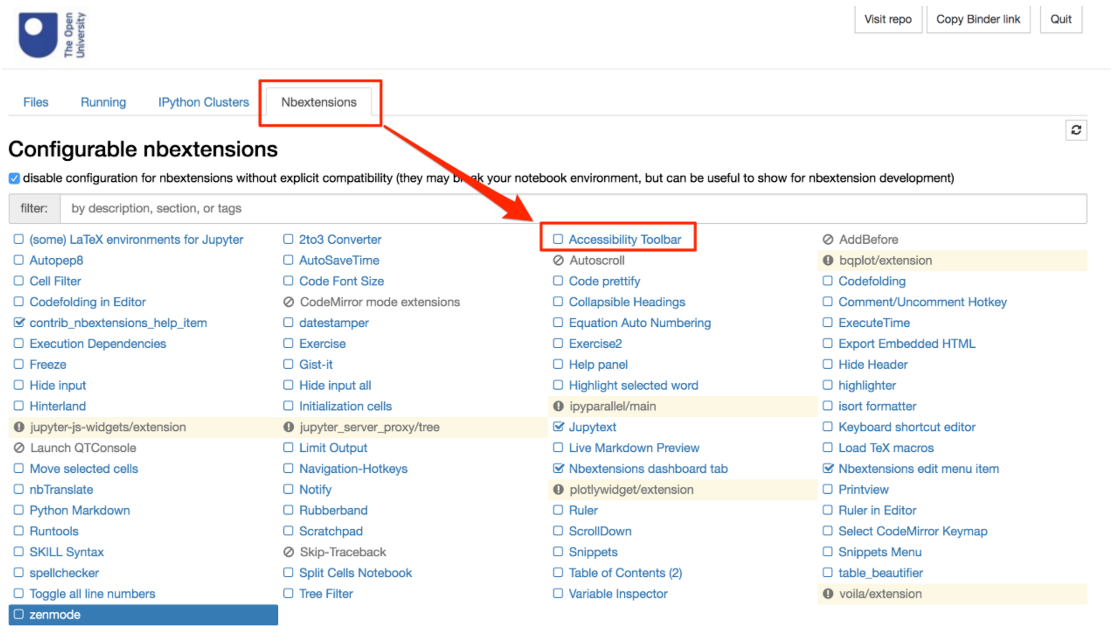
Figure 6.2: The Jupyter notebook dialogue box for editing keyboard shortcuts
Screenshot of the Jupyter notebook settings dialog for editing keyboard shortcuts. Examples of several commands for which shortcuts are available are listed, along with the associated keyboard shortcut, if defined. Example commands listed include: “shutdown kernel”, “confirm shutdown kernel”, “restart kernel”, “restart kernel and ruin all cells”, “reastart kernel and clear output”, “interrupt kernel” and “run cell”.

6.3 Accessibility Toolbar

The Jupyter environment includes an experimental accessibility toolbar extension that allows you to control the presentation style of the Jupyter notebook; for example, you can change the font style, size and spacing, the notebook background colour, and so on.

6.3.1 Enabling the Accessibility Toolbar

If the accessibility toolbar extension is not displayed, you will need to enable it. You can do this from the `Nbextensions` tab on the notebook home page (you may need to toggle the extension configurator *disable configuration* option first): tick the *Accessibility Toolbar* extension to enable the toolbar (Figure 6.3). When you open a new notebook, the toolbar should be displayed.



Configurable nbextensions

☑ disable configuration for nbextensions without explicit compatibility (they may break your notebook environment, but can be useful to show for nbextension development)

filter: by description, section, or tags

☐ (some) LaTeX environments for Jupyter	☐ 2to3 Converter	☐ Accessibility Toolbar	⊘ AddBefore
☐ Autopep8	☐ AutoSaveTime	⊘ Autoscroll	● bqplot/extension
☐ Cell Filter	☐ Code Font Size	☐ Code prettify	☐ Codefolding
☐ Codefolding in Editor	⊘ CodeMirror mode extensions	☐ Collapsible Headings	☐ Comment/Uncomment Hotkey
☑ contrib_nbextensions_help_item	☐ datestamper	☐ Equation Auto Numbering	☐ ExecuteTime
☐ Execution Dependencies	☐ Exercise	☐ Exercise2	☐ Export Embedded HTML
☐ Freeze	☐ Gist-it	☐ Help panel	☐ Hide Header
☐ Hide input	☐ Hide input all	☐ Highlight selected word	☐ highlighter
☐ Hinterland	☐ Initialization cells	● ipyparallel/main	☐ isort formatter
● jupyter-js-widgets/extension	● jupyter_server_proxy/tree	☑ Jupytext	☐ Keyboard shortcut editor
⊘ Launch QTConsole	☐ Limit Output	☐ Live Markdown Preview	☐ Load TeX macros
☐ Move selected cells	☐ Navigation-Hotkeys	☑ Nbextensions dashboard tab	☑ Nbextensions edit menu item
☐ nbTranslate	☐ Notify	● plotlywidget/extension	☐ Printview
☐ Python Markdown	☐ Rubberband	☐ Ruler	☐ Ruler in Editor
☐ Runtools	☐ Scratchpad	☐ ScrollDown	☐ Select CodeMirror Keymap
☐ SKILL Syntax	⊘ Skip-Traceback	☐ Snippets	☐ Snippets Menu
☐ spellchecker	☐ Split Cells Notebook	☐ Table of Contents (2)	☐ table_beautifier
☐ Toggle all line numbers	☐ Tree Filter	☐ Variable Inspector	● voila/extension
☐ zenmode			

Figure 6.3: The Jupyter Nbextensions tab showing the Accessibility Toolbar extension
Screenshot of the Jupyter Nbextensions tab listing all the installed ecxtensions and highlighting the option for enabling the Accessibility Toolbar extension.

Check the accessibility toolbar documentation for more information.

All of the styles are saved into local storage when refreshing the page. This means that if you use notebooks on different servers with the same browser, the same accessibility settings will be applied to notebooks on all servers within which you have enabled the accessibility extension.

6.3.2 Controlling colours and fonts using the Accessibility Toolbar

If you wish to change the way in which text is displayed in order to improve readability, the accessibility toolbar allows you to select the font style, size and colour. You can also modify the line spacing and spacing between individual characters as shown in Figure 6.4.

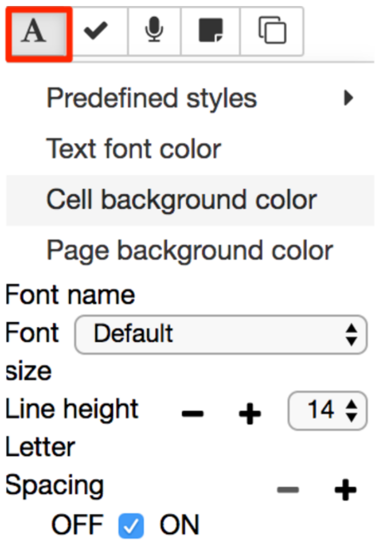


Figure 6.4: Accessibility toolbar font style options
Font style options offered by the accessibility toolbar.

The font style applies to *all* text elements within the notebook itself. This includes the contents of Markdown (text) cells, code cells and code cell outputs.

The toolbar can also be used to control the notebook's background colour and the cell background colour.

You can also save a style you have defined from the *Add new style…* option in the *Predefined styles* menu. Once saved, it will be added to the menu list so you can apply it as required.

6.4 Other assistive software

Please contact your tutor if you discover that the material does not work with a particular screen reader or dictation system that you would typically expect to be able to use.

CHAPTER SEVEN

JUPYTER NOTEBOOK GOTCHAS

In computer slang, a "gotcha" often refers to a situation where you may be caught out by expecting a system to behave in one way when it actually behaves in another. There are several "gotchas" that may catch out the unwary when using Jupyter notebooks.

1. Typically, code cells are intended to be executed in linear order as you work through a notebook. However, code cells can be executed in any order. Take care not to execute cells out of order. If your code is not executing as expected, and you think it may be because of out of order code execution, restart the notebook kernel and execute the cells again in the correct order up to the cell you are working on.
2. A displayed cell output may not reflect the current state of the code environment. If a cell is used to display the output of a variable, if the value of the variable is changed by code executing later in the notebook, the previously displayed is not updated. The output is the value of the variable at the time the cell generated the output.
3. When you open a *new* notebook, the notebook server starts a new Jupyter kernel. However, opening a previously opened notebook does not necessarily create a new notebook kernel. If the notebook tab was closed without stopping the kernel, the previous kernel may still be running when the notebook is reopened. You can restart a kernel at any point to ensure you are working with a fresh kernel. The kernel will be restarted in a state where none of the code cells have run.
4. If you hibernate your computer and then wake it up again, any running Jupyter kernel processes should just hibernate and still retain their previous state. If you stop a notebook, either from the notebooks folder home page or by stopping and restarting the VCE, then the Jupyter kernel process will be stopped and all states (as far as the process is concerned) will be lost.
5. If you open the same notebook in more than one browser tab (which is not necessarily a sensible thing to do!), all copies of the notebook will access the same Jupyter kernel. You can see this if you set a variable value in a notebook in one tab: you will be able to display the value in the notebook in the other tab. (Run a cell containing `a=5` in one tab, and run a cell containing print(a) in a second tab the same Jupyter kernel process is visible in both instances of the notebook.) If one of the tabbed copies of the notebook is saved, then the other may pop up an alert noticing the difference and ask what you want to do about it. Generally, it's a good idea to avoid this problem by only opening a given notebook in a single tab.

6. If you open a notebook with a new kernel, the values of any previously run cell outputs have no necessary relation with the current state of the kernel. Clear all the cell outputs to prevent any confusion as to whether the cell output was generated in the current notebook session.
7. It is possible that you may attempt to close the browser tab for a notebook without the content changes having been saved. The browser will usually display a warning and offer you the option of staying on the page, or leaving the page – this warning will differ depending on your host browser, but will say something like ‘Are you sure you want to leave this page?’. This is a sign that the most recent changes to the page haven’t been saved – usually you want to stay on the page, click the save icon, then close the tab.

CHAPTER

EIGHT

ADDITIONAL SUPPORT

If you encounter any problems installing the VCE or accessing the services that it provides, or have problems running provided code or your own code, the first thing to remember is **don't panic**: in many cases, you may be able to fix the problem yourself; in others, a readily available helping hand should be able to resolve your problem.

In general, you may find it useful to work through the problem in the following way:

1. Check any error messages, if appropriate. Error messages often contain a clue as to what caused them. If you are using a terminal to issue commands, or writing computer code, a common problem is a command not being recognised because of a typographical error. Computers are very literal in what they do, so a misplaced comma or a single character out of place may be all that needs fixing.
2. Check the module forums to see if there are any official announcements relating to the same problem, or whether any other students have posted similar issues. Many forums are moderated and the forum moderators will answer any questions they can, or pass them on to the module team if they can't. Appropriate responses will then be posted to the forum.
3. If a problem similar to yours has already been reported, but the suggested solution does not work for you, reply to the suggestion. In your reply, explain what you tried and what did or did not happen, including any error messages, if available, either as copied text or as a screenshot. Provide details of your operating system, if relevant. Sometimes you might find that trying to explain the issue identifies the solution. In such a case, if you think the problem may be a common one, you may want to post a message to the forum explaining the error you encountered and how you fixed it.
4. If your problem has not already been identified, post a message to the 'Jupyter notebooks' forum. Give the message a clear title that identifies the problem. Include in the body of the message a description of what you did and what did or did not happen, including any error messages, if available, either as copied text or as a screenshot. Once again, provide details of your operating system and/or browser, if appropriate. As before, you may find that trying to explain the problem reveals the solution. If you think that sharing the issue — and the solution you discovered — may be of some help to others, please do so via the appropriate forum.
5. If you are struggling with identifying what the problem is and would find it useful to talk to someone, get in contact with:

- the OU Computing Helpdesk, for general IT and software installation enquiries
- your tutor, by phone or email or other agreed contact method for issues specifically about taught content and practical activities within the module.

To give yourself the best chance of resolving any issue in a timely fashion, *plan ahead*. The assignments require you to make use of the VCE, so don't leave it until the last minute to try the VCE for the first time.

More specific guidance for working through issues related to various software problems can be found in *Troubleshooting*.

CHAPTER

NINE

TROUBLESHOOTING

While we have tried to ensure that the software installation process and all the software-related activities run smoothly, there is always a chance that you may encounter a problem or issue with the software.

The section *Additional support* describes a general strategy for working through problems or raising technical issues. This section describes more specific guidance for working through issues with particular software elements.

Optional content

You shouldn't need to work though the following unless there are specific problems that you have encountered when working with the VCE.

9.1 Problems installing and/or running a local VCE

When you first access or install the VCE, we suggest that you run through the `READ_ME_FIRST.ipynb` notebook as soon as you can. If you have problems installing Docker or the Docker container / VCE, please be sure to include details of your operating system when requesting assistance via the module forum.

You can check whether your VCE container with name `CONTAINERNAME` (for example, `m348vce`) is currently running from the list of running containers displayed in the Docker Desktop, or by entering the command `docker ps` in a terminal / command prompt.

If the container is not shown as running in the Docker Desktop, or you cannot see it reported as *Up* for a certain period of time in the `docker ps` listing, or the services appear not to be running (your browser doesn't connect to the service after you have entered the appropriate web address then *restart* the VCE from the Docker Desktop or by issuing the terminal command `docker restart CONTAINERNAME`.

9.2 Issuing Docker commands on the command line

9.2.1 Docker in Microsoft Windows

To change directory to the desired location, open the command prompt and use the `CD` command, followed by the path to the directory (i.e. folder) you want to move to. To display the name of the current directory, enter `CD` without any parameters. To list the contents of the current directory, use the `DIR` command.

9.2.2 Docker with Mac and Linux

To change directory to the desired location, open a terminal and use the `cd` command, followed by the path to the directory (i.e. folder) you want to move to. To display the name of the current directory, enter `pwd` without any parameters. To list the contents of the current directory, use the `ls` command.

9.3 Docker Error messages

When running the `docker run` command, if you see the error message:

```
docker: Error response from daemon: Conflict. The container name-
CONTAINERNAME is already in use by container "...". You have to
remove (or rename) that container to be able to reuse that name.
```

you have already created a container with that container name. You should be able to restart it using the command `docker restart CONTAINERNAME` or from the Docker Desktop.

If you need to create a new instance of the container, delete the current instance by running the command `docker rm -f CONTAINERNAME` or using the Docker Desktop, and then try running the `docker run` command again.

9.4 Recovering a Docker container after sleep

If your computer goes to sleep, the container will also be hibernated. When you wake your computer, the container and the services it is running should also wake up, although you may have to restart any Jupyter notebook kernels you left running. In rare cases, you may find that the container has got stuck somehow. In such a case, check whether the VCE container is currently running by entering the command `docker ps` in a terminal / command prompt and looking for the appropriately named container.

If the container is not running (you cannot see it reported as 'Up' for a certain period of time in the `docker ps` listing) or the services appear not to be running (your browser doesn't connect to the service after you have entered the appropriate address, then *restart* the VCE from the Docker Desktop containers display or by issuing the terminal command `docker restart CONTAINERNAME`.

9.5 Taking drastic action

Take a backup first

You might want to download your files or make a backup of the shared folder before attempting the following.

If for any reason you need to delete a VCE container instance and create a replacement container from the original image, or update the Docker image and create a newly updated container instance, stop and delete the container using the Docker Desktop, or run the following command to stop and delete the container:

```
docker rm --force CONTAINERNAME
```

You should now create a new container using Docker Desktop or by executing the original `docker run` command again.

Deleting the container will *not* delete any of the contents of the shared folder.

9.6 Problems accessing the local VCE in your browser

When using the local VCE, if you cannot see the notebooks folder home page in your browser, the first thing to check is that the VCE is running from the Docker Desktop container listing. Alternatively, on the command line, run the command `docker ps --filter "name=CONTAINERNAME"` or the more general `docker ps` to see whether the container is running. If the container is *not* running, then restart it from the Docker Desktop or by issuing the command `docker restart CONTAINERNAME`.

You can then check the services have been started and where the services are running as described below.

9.7 Problems arising from local VCE notebook permissions

If you encounter permissions-based problems when trying to move files into the shared folder or creating files or folders from the notebook server homepage, then check the file permissions. From a notebook code cell, or from a terminal running inside the container, list the directories set in your home directory (~), along with their permissions, by running the command:

```
ls -l ~
```

If you notice that any of the files have the user:group field set to `root` rather than the `ou` user, reset the permissions by running the following command from PowerShell (Windows) or a terminal (Mac/Linux) on your host computer:

```
docker container exec -itu 0 CONTAINERNAME chown -R ou:users /home/
ou/
```

If you still see an error, restart your computer. You will also need to (re)start the container, either directly from the Docker Desktop, or in PowerShell/your terminal by running the command:

```
docker restart CONTAINERNAME
```

On Windows, if you create a shared folder then run the docker run command in the same directory, you may see the error message:

```
docker: Error response from daemon: mkdir C:\\WINDOWS\\system32\
\SHAREDFOLDERNAME: Access is denied.
```

If you see this error message, delete the shared folder and execute the docker run command again. This will automatically recreate the folder.

9.8 Accessing a terminal command-line interface within the VCE

For many modules, you should not need to access the VCE command line.

However, if you find you do want to issue a command-line command or gain access to a command-line interface or terminal within the virtual machine, there are three main ways of doing this from inside the container:

- Within a notebook code cell, in the first line of the cell enter the command:

```
%%bash
```

Any additional code lines you enter into the cell will be interpreted as shell commands and executed as such when you run the cell.

- Within a notebook code cell, prefix the command-line command you want to run with an exclamation mark (!). For example, to list the contents of the current directory, run the Linux/Unix ls command:

```
! ls
```

- Use the Jupyter interactive shell. From the notebooks folder home page, create a new terminal by selecting *Terminal* from the *New* drop-down menu on the right-hand side as shown in Figure 9.1.

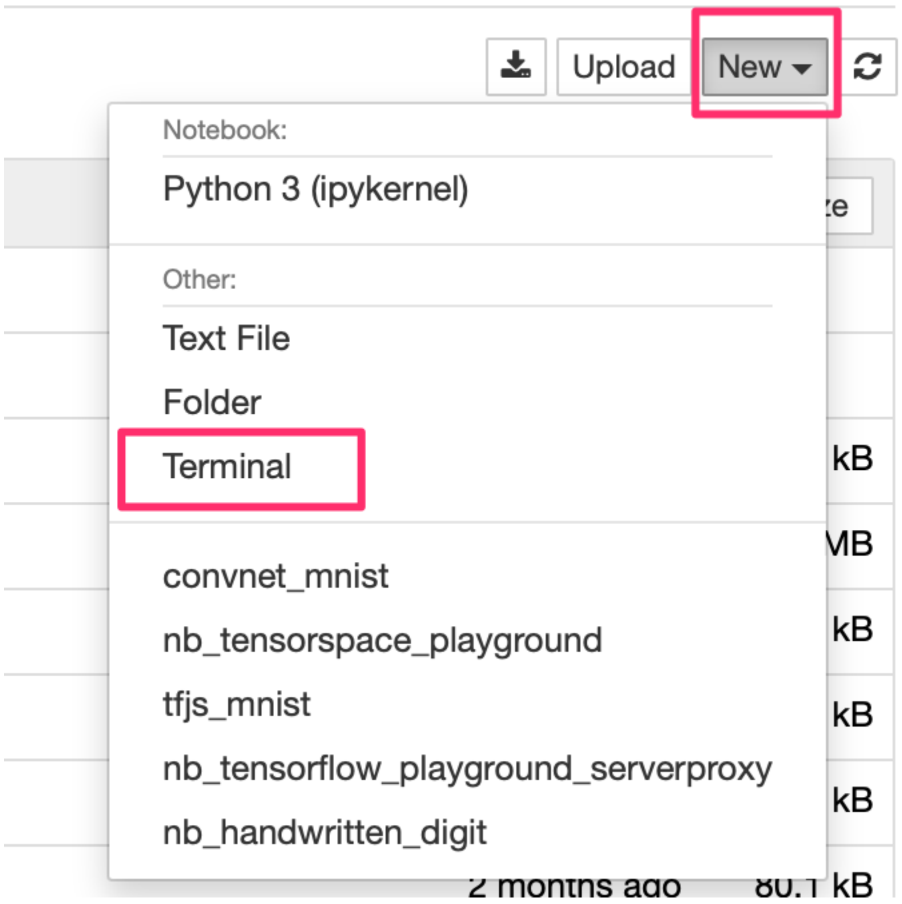


Figure 9.1: The "New" drop-down menu
A screenshot of part of the notebooks folder home page showing three buttons — 'Download as zip' (indicated as a down arrow above a horizontal bar), 'Upload' and 'New' — and a refresh button. The 'New' button has been pressed, revealing a drop-down menu with various items'. The item 'Terminal' has been highlighted.

- Via a command line / terminal / command prompt on your host computer, change directory to the shared folder and then run:

```
docker exec -it CONTAINERNAME /bin/bash
```

If you are running the local VCE, you can also launch a terminal that is connected to the container from the Docker Desktop running container page. If you need to enter the container as the root user, you can do so by running the following command on your host computer:

```
docker container exec -itu 0 containername /bin/bash
```

The Docker container that houses the VCE is running a version of Linux.

The table below provides a quick summary of Linux/Unix commands you might find useful.

Command	Description
`pwd`	Show the path to the current working directory (folder).
`cd PATH`	Change directory to the specified path (where `..` signifies 'parent of' and a single `.` represents the current directory).
`ls`	List files and folders.
`ls /home/ou`	List contents of the specified directory. Inside the VCE, the `/home/ou` path is the recommended default path for the shared folder.
`head PATH/FILENAME`	Preview the first 10 lines of the specified file.
`tail -n 20 PATH/FILENAME`	Preview the last 20 lines of the specified file.
`man COMMAND`	Display manual pages for *command*, e.g. `man pwd`.
`wget URL`	Download the file from the specified URL to the current directory.
`tar -xvf FILENAME`	Uncompress and unarchive files from a *.tar.gz* or *.tgz* file.

9.9 Problems arising from working with large notebooks

If you have tried to display a large amount of data in a notebook cell then you may find that your browser struggles to display the notebook.

You should first attempt to clear all the output cells using the notebook menu *Cell* > *All Output* > *Clear*. If you are entirely unable to run the notebook to access the menu, first try closing any other notebooks you have open then try opening the problem notebook. If that doesn't allow the notebook to open, then using a command-line command you can run the problem notebook through a separate process that will clear all the cell outputs.

Open a terminal in the VCE. Then change directory (`cd`) to the shared folder, and issue the following command (all on one line):

```
jupyter nbconvert --to notebook --ClearOutputPreprocessor.
enabled=True YOURNOTEBOOK.ipynb
```

By default, the clean notebook will be named `YOURNOTEBOOK.nbconvert.ipynb`. To clean the notebook and retain the same filename, add the flag `--inplace` to the command line:

```
jupyter nbconvert --inplace --to notebook
--ClearOutputPreprocessor.enabled=True YOURNOTEBOOK.ipynb
```

The notebook server may also run into memory problems if you have a large number of notebooks open. Try stopping all the notebooks and then restarting the notebook you are currently working on. (You will need to rerun the code cells to restore the state of the variables.)

9.10 Problems associated with running out of memory

If in a long running container, if you close a notebook but do not stop the associated kernel, you may end up with a large number of still running kernels even if you do not have any notebook tabs open in your browser. You can find a list of running notebooks from the *Running* tab on the notebook homepage.

You can also see an indication of which notebooks still have running kernels associated with them when viewing a file listing on the notebook home page, Figure 19.

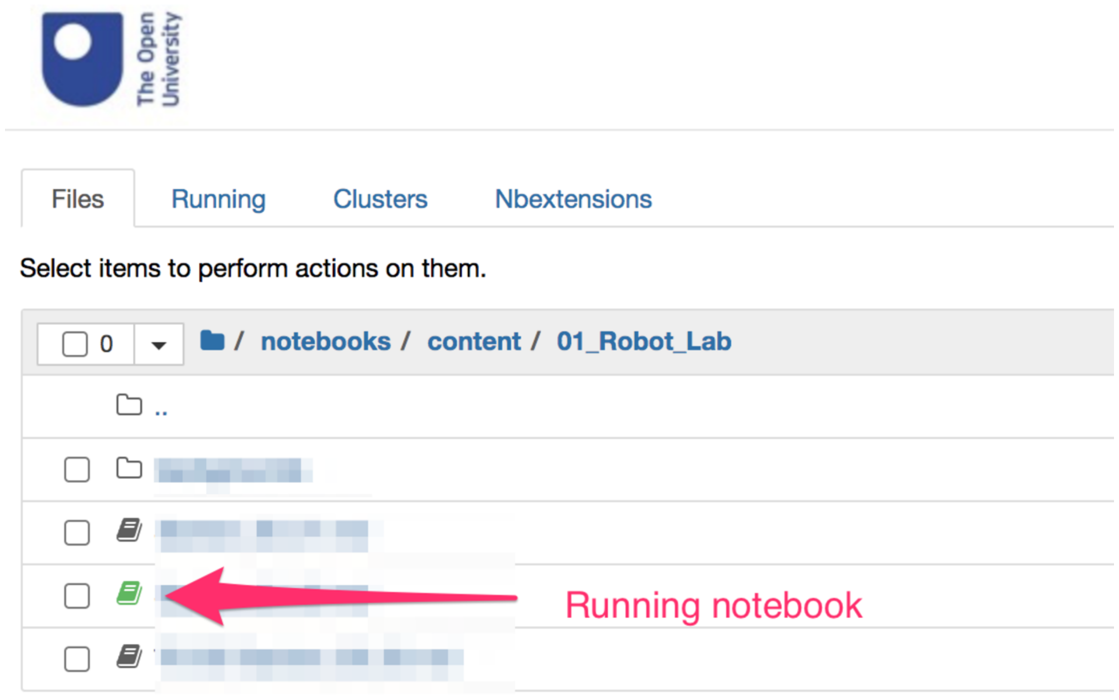

The Open University
Files
Running
Clusters
Nbextensions
Select items to perform actions on them.
0
/ notebooks / content / 01_Robot_Lab
..
Running notebook

Figure 9.2: Figure 20 An example of the notebooks folder home page. The actual directory file path and file listing is indicative only.
DESCRIPTION: A screenshot of a web browser showing the 'Files' tab of the Jupyter notebook environment. A number of items are shown pixellated, but the one with a green icon has an arrow pointing to it labelled 'Running notebook'.

If you do find you have a lot of running kernels without associated open notebooks, stop them from the *Running* tab.

9.11 Problems running notebook code

If you have problems running code in the module notebooks, or running your own code, read any error messages carefully to see if they point to the cause of the problem. You may also find it useful to check the relevant documentation, or run a web search using key elements of any error messages that occur.

Technical Q&A sites such as Stack Overflow contain answers to a wide variety of '*How do I … ?*' style questions and are often well worth exploring. Results from searches on Stack Overflow can also be limited to relevant questions by adding appropriate programming language tags to your query, such as *python* or *R*.

If you continue to have problems, check the module forums to see if anyone else has encountered — and resolved — the same issue. You may also ask for support in module forums or from your tutor.

The OU Computing Helpdesk will not be able to help with detailed technical queries arising from the module practical activities, so please limit any module-content related enquiries to the module forums or your tutor. The OU Computing Helpdesk can help with more general computing matters, including some support for installation problems that might occur.

9.12 Finding version numbers for software in your VCE

There may be occasions, such as when troubleshooting or debugging problems, when you are asked to confirm which versions of software applications your are running or programming packages you have installed in your VCE. You will be provided with guidance on how to find the required version numbers.

If you are running into repeated problems with an environment inside a local VCE, report the digest number for the image used to generate the container. You can find this by running the command `docker images --digests` from the command line on your computer. Look for the line corresponding to the `ousefulcoursecontainer/ou-m348:23j` image: the digest is the number starting with the prefix: `sha256:`.

9.13 Windows performance issues

You may find that performance of your Windows computer degrades when running Docker under WSL2.

This is caused by WSL2 appropriating too much computational resource from your computer. The solution is to create a file called `.wslconfig` in `C:\\Users\your-username\` containing the lines:

```
[wsl2]
memory=4GB # Limits VM memory in WSL 2 to 4 GB
processors=2 # Makes the WSL 2 VM use two virtual processors
```

Using an account with admin rights, from a PowerShell command line, restart WSL2 by entering:

```
Restart-Service LxssManager
```

You should find that setting `memory=2GB` is enough to run the container although you may find things run more smoothly by setting `memory=4GB`.

CHAPTER

TEN

APPENDIX — VCE TECHNICAL ARCHITECTURE

Many Open University modules require the use of third-party software tools, applications and programming environments. Through the use of virtual computing environments (VCEs), we are able to provide identical environments to students using remotely accessed VCEs hosted on the Open University's Compute Home servers or running locally on your own computer.

Optional content

You shouldn't need to work though the following unless there are specific problems that you have encountered when working with the VCE.

10.1 Docker Containers and the VCE

The VCE runs in a virtualised container under the Docker application as a *guest* Linux (Ubuntu) operating system on a host computer such as Compute Home or your own computer. The containers typically operate in a 'headless' mode (that is, without a graphical desktop interface) and run a variety of application services. The applications are exposed as interactive, graphical web applications via an HTTP interface that you can access through a web browser.

The local and hosted VCEs both runs as a single user environment. The hosted VCE launches a separate containerised environment for each student using a JupyterHub multi-user server.

Persistent file storage is available on both the hosted and local VCE.

In the hosted VCE, files are saved to a persistent personal file storage area that is accessible through the hosted VCE. Experimental extensions are available for some Jupyter environments and in some browsers that can map a selected folder on a user's own computer into the browser and execute notebooks in that folder using the remote VCE.

If a local VCE is created, it can be configured to share a folder with the host computer. The contents of this folder are visible to both the host computer and the local VCE (that is, the guest machine); they also remain visible on the host even if the guest machine is not running. The shared folder thus provides a way of passing files from your host machine into the local VCE, as well as getting files (such as backup files) out of the local VCE and onto your host computer. Figure 10.1 broadly describes the architecture of a local VCE and its relationship with your host operating system.

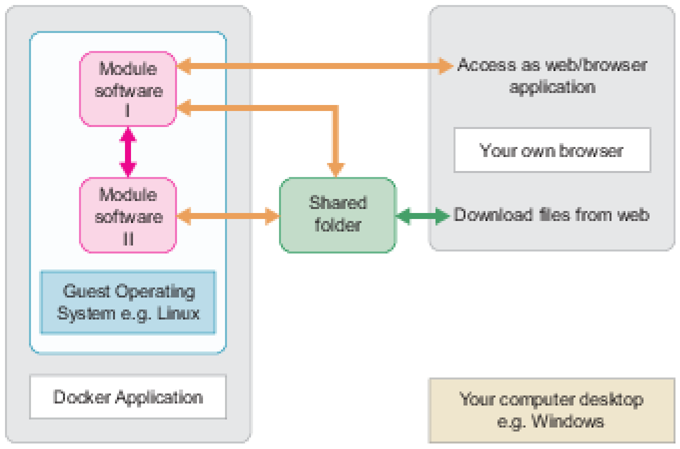


Figure 10.1: Overview of the architecture of a virtual computing environment
The architecture of a VCE is shown as a block diagram – showing the embedding and linking between components. The architecture shows that your computer, for example a Windows PC, directly runs the Docker application, the shared folder and your browser. The Docker application creates a Docker container that contains a guest Linux operating system that runs various pieces of module software. You can use your browser to access web applications and download files from the web, but in addition it can also access the module software on the guest operating system. The shared folder is shown as being accessed from the containerised virtual computing environment software and from the browser on the host. The shared folder is accessible from your browser (so you can upload files from the web and place them in the shared folder) and from the guest operating system (so the module software can read and write files in the shared folder).

Many VCEs provide access to the Jupyter classic notebook or Jupyter Lab interactive environment via a web browser. The Jupyter notebook user interfaces provides an interactive notebook style environment for writing, executing and capturing the output of executable program code using a wide variety of programming languages, such as Python or R.

Within the VCE, applications publish services on port numbers associated with the ‘localhost’ network *inside* the virtual environment. Docker then *forwards* the traffic on published ports to another set of port numbers that are visible on the localhost network relative to the host. The localhost network within the VCE is *not* the same network as the localhost network on the host. This is shown in Figure 18.

VCE applications are typically accessed from a web browser. When using the remote VCE, you will be automatically forwarded to the correct URL from Compute Home. In the local VCE, using the suggested default settings, the notebooks can be accessed via a URL that addresses a particular port on the computer’s own localhost internal network with numerical address 127.0.0.1.

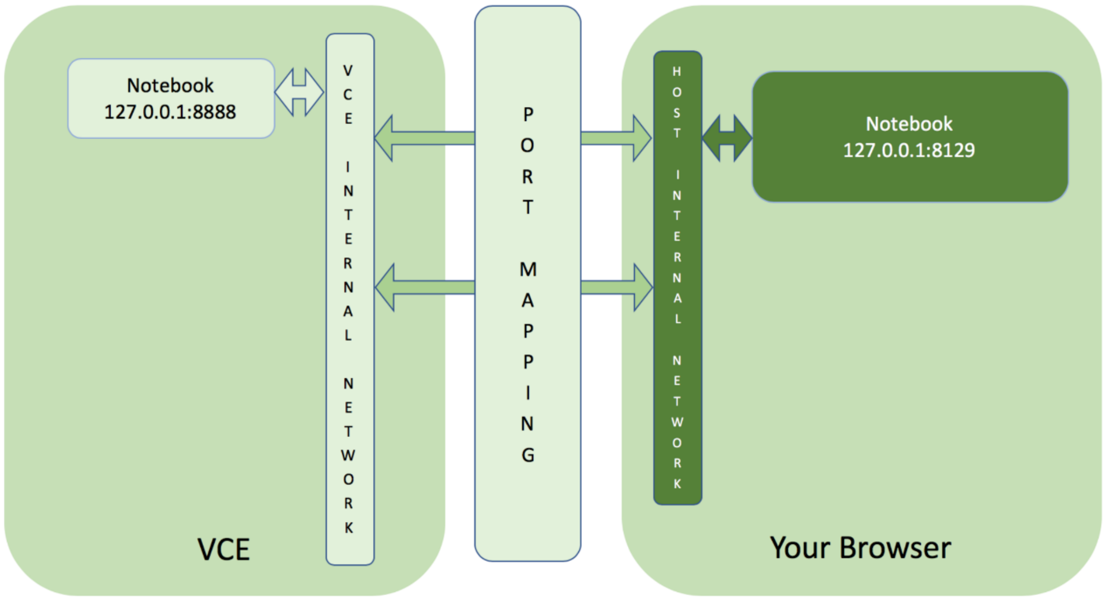


Figure 10.2: Port mapping from the Docker container to the host network
A block diagram showing the relationship between software and the network ports used in the virtual computing environment (VCE), and the ports accessible on the host machine through your browser. The virtual computing environment and the host machine are connected by port-mapping software that translates between port addresses in each machine. The virtual computing environment is shown on the left-hand side with an internal network, with the address 127.0.0.1, to which various software processes are connected through specific local port numbers – these are the guest ports of interest on TM129. The notebook server is on port 127.0.0.1:8888. Each port connects to the VCE internal network and through that to the port-mapping software, which in turn connects to the host internal network, shown on the right-hand side. On the host computer, the host internal network is shown as connecting to software processes in the browser through the ports; the software shown here is the notebooks on 127.0.0.1:8129.

10.2 Creating your own version of a module VCE

Many Open University VCEs incorporate several inter-dependent software packages and applications. These self-contained virtual computing environments have been developed to provide a ‘ready-to-use’ environment that contain all the software you need to complete your studies within a particular module and can typically be run via the hosted Compute Home service, or on your own computer using a Docker container launched from a prebuilt Docker image.

If you want to build your own version of the environment **— which is *not* required for the module, and is *not* recommended in most cases —** please contact the module team via the module forums. Note that the module team are unlikely to be able to support students creating their own environments although they may be able to indicate what the installation requirements are.