Exercise Worksheet

[www.vomtom.at](http://www.vomtom.at)

# From the Course: Understanding Docker Run, Dockerfile, Docker-Compose for Beginners

## Using PHP with volume mounting in docker images step by step

docker run -it --rm --name my-running-script php:7.2-cli /bin/bash

* Downloads the official php 7.2 cli image
* Runs a shell inside

php -m

* Get the installed modules

php -i

* Get the config of the PHP cli

echo '<? echo "test text\n";' > ~/index.php

* This will write <? echo “text text\n”; into the index.php file in the home directory of root
* Let’s execute this

php ~/index.php

* Should print now “test text”

exit

* Will end the container, but will also remove the container (including our index.php) because of the --rm flag
* This is where mounting an external host directory comes in handy

docker run -it --rm -v ${PWD}:/myfiles --name my-running-script php:7.0-cli /bin/bash

* Now we are running the php 7.0, not the php 7.2 cli
* And we are mounting the root directory into the /myfiles directory inside the container

echo '<? echo "test text\n";' > /myfiles/index.php

* This creates the same index.php, but this time inside /myfiles.
* Observe the host directory

php /myfiles/index.php

* This should run index.php with the php 7.0 interpreter

exit

* We can safely exit again, because our file is securely stored on the host. It will not be destroyed with the container upon exiting

docker run -it --rm -v ${PWD}:/myfiles --name my-running-script php:7.2-cli /bin/bash

* Let’s re-run the 7.2-cli container
* This time with the same directory mounted in /myfiles

php /myfiles/index.php

* This allows you to execute the same script with another PHP version

exit

* Let’s exit the container

docker run -it --rm -v ${PWD}:/myfiles -w /myfiles --name my-running-script php:7.2-cli php index.php

* Will directory run index.php and output the “test text” and exit again
* No need to actually enter the container
* -w flag again sets the working directory to /myfiles