1. **What is a container?**
2. Using docker run -d -p 8080:80 docker/welcome-to-docker to fire up my first container

文本

AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

1. Then I can visit [http://localhost:8080](http://localhost:8080/) in my browser.

图形用户界面

AI 生成的内容可能不正确。

1. Using docker stop pedantic\_davinci to stop my first container pedantic\_danvici. 文本

   AI 生成的内容可能不正确。
2. **What is a image?**
3. Open a terminal and search for images using the docker search command:

docker search docker/welcome-to-docker

文本

AI 生成的内容可能不正确。

1. Pull the image using the [docker pull](https://docs.docker.com/reference/cli/docker/image/pull/) command.

docker pull docker/welcome-to-docker

文本

AI 生成的内容可能不正确。

1. List your downloaded images using the [docker image ls](https://docs.docker.com/reference/cli/docker/image/ls/) command:

docker image ls

**文本

AI 生成的内容可能不正确。**

1. List the image's layers using the docker image history command:

docker image history docker/welcome-to-docker

电脑屏幕截图

AI 生成的内容可能不正确。

1. **What is Docker Compose?**
2. Download and install Docker Desktop.
3. Open a terminal and clone this sample application.

git clone [git@github.com:dockersamples/todo-list-app.git](mailto:git@github.com:dockersamples/todo-list-app.git)

文本

AI 生成的内容可能不正确。

1. Navigate into the todo-list-app directory:

cd todo-list-app

文本

AI 生成的内容可能不正确。

1. Use the [docker compose up](https://docs.docker.com/reference/cli/docker/compose/up/) command to start the application:

docker compose up -d --build

文本

AI 生成的内容可能不正确。

1. With everything now up and running, you can open [http://localhost:3000](http://localhost:3000/) in your browser to see the site. Feel free to add items to the list, check them off, and remove them.

图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

1. **Publishing and exposing ports**
2. Download and install Docker Desktop.
3. In a terminal, run the following command to start a new container:

docker run -d -p 8080:80 docker/welcome-to-docker



1. Verify the published port by going to the **Containers** view of the Docker Desktop Dashboard.

图形用户界面, 应用程序, 电子邮件

AI 生成的内容可能不正确。

1. Open the website by either selecting the link in the **Port(s)** column of your container or visiting [http://localhost:8080](http://localhost:8080/) in your browser.

图形用户界面

AI 生成的内容可能不正确。

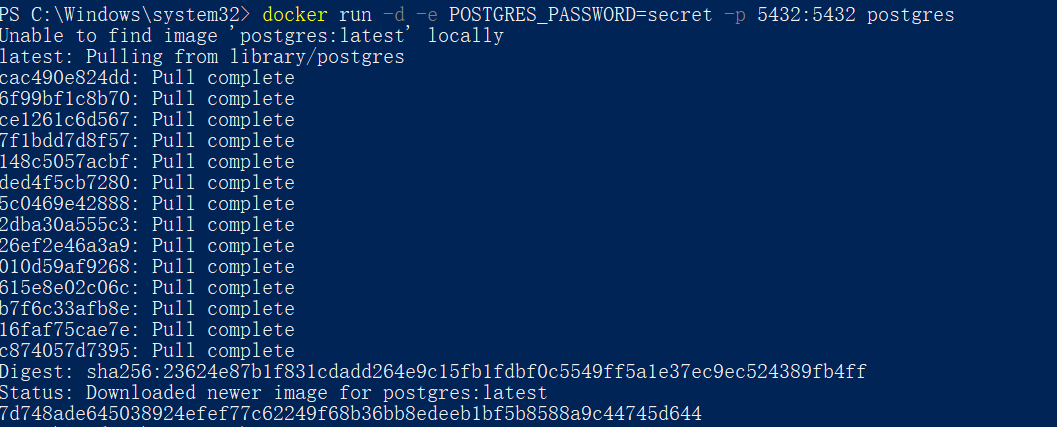
1. **Overriding container defaults**

**Run multiple instances of the Postgres database**

1. Start a container using the Postgres image with the following command:

docker run -d -e POSTGRES\_PASSWORD=secret -p 5432:5432 postgres

This will start the Postgres database in the background, listening on the standard container port 5432 and mapped to port 5432 on the host machine.



1. Start a second Postgres container mapped to a different port.

docker run -d -e POSTGRES\_PASSWORD=secret -p 5433:5432 postgres

This will start another Postgres container in the background, listening on the standard postgres port **5432** in the container, but mapped to port **5433** on the host machine. You override the host port just to ensure that this new container doesn't conflict with the existing running container.



图形用户界面, 应用程序

AI 生成的内容可能不正确。

**Run Postgres container in a controlled network**

1. Create a new custom network by using the following command:

docker network create mynetwork



1. Verify the network by running the following command:

docker network ls

手机屏幕的截图

AI 生成的内容可能不正确。

1. Connect Postgres to the custom network by using the following command:

docker run -d -e POSTGRES\_PASSWORD=secret -p 5434:5432 --network mynetwork postgres

This will start Postgres container in the background, mapped to the host port 5434 and attached to the mynetwork network. You passed the --network parameter to override the container default by connecting the container to custom Docker network for better isolation and communication with other containers. You can use docker network inspect command to see if the container is tied to this new bridge network.

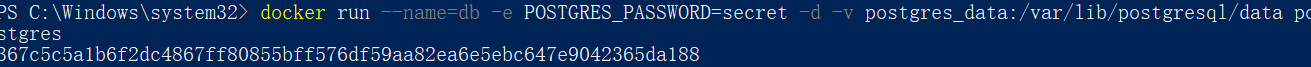
 图形用户界面, 应用程序

AI 生成的内容可能不正确。

1. **Persisting container data**
2. Download and install Docker Desktop.
3. Start a container using the Postgres image with the following command:

docker run --name=db -e POSTGRES\_PASSWORD=secret -d -v postgres\_data:/var/lib/postgresql/data postgres

This will start the database in the background, configure it with a password, and attach a volume to the directory PostgreSQL will persist the database files.

 图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

1. Connect to the database by using the following command:

docker exec -ti db psql -U postgres

文本

AI 生成的内容可能不正确。

1. In the PostgreSQL command line, run the following to create a database table and insert two records:

CREATE TABLE tasks (

id SERIAL PRIMARY KEY,

description VARCHAR(100)

);

INSERT INTO tasks (description) VALUES ('Finish work'), ('Have fun');

图形用户界面, 文本

AI 生成的内容可能不正确。

1. Verify the data is in the database by running the following in the PostgreSQL command line:

SELECT \* FROM tasks;

文本

AI 生成的内容可能不正确。

1. Exit out of the PostgreSQL shell by running the following command:

\q

图形用户界面, 文本

AI 生成的内容可能不正确。

1. Stop and remove the database container. Remember that, even though the container has been deleted, the data is persisted in the **postgres\_data** volume.

docker stop db

docker rm db

1. Start a new container by running the following command, attaching the same volume with the persisted data:

docker run --name=new-db -d -v postgres\_data:/var/lib/postgresql/data postgres

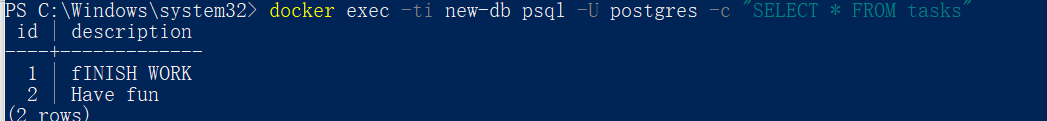
You might have noticed that the POSTGRES\_PASSWORD environment variable has been omitted. That’s because that variable is only used when bootstrapping a new database.

图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

1. Verify the database still has the records by running the following command:

docker exec -ti new-db psql -U postgres -c "SELECT \* FROM tasks"



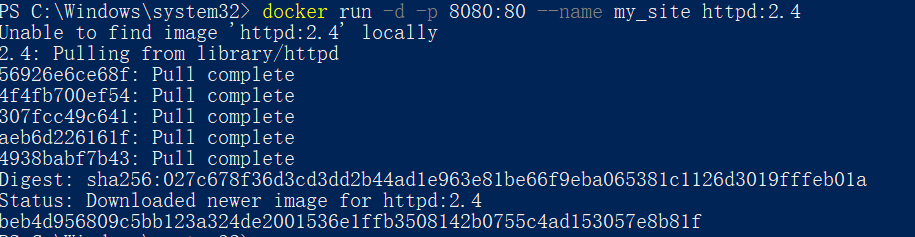
1. **Sharing local files with containers**

**Run a container**

1. Download and install Docker Desktop.
2. Start a container using the httpd image with the following command:

docker run -d -p 8080:80 --name my\_site httpd:2.4

This will start the httpd service in the background, and publish the webpage to port 8080 on the host.

图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。

1. Open the browser and access http://localhost:8080 or use the curl command to verify if it's working fine or not. 图形用户界面

   AI 生成的内容可能不正确。

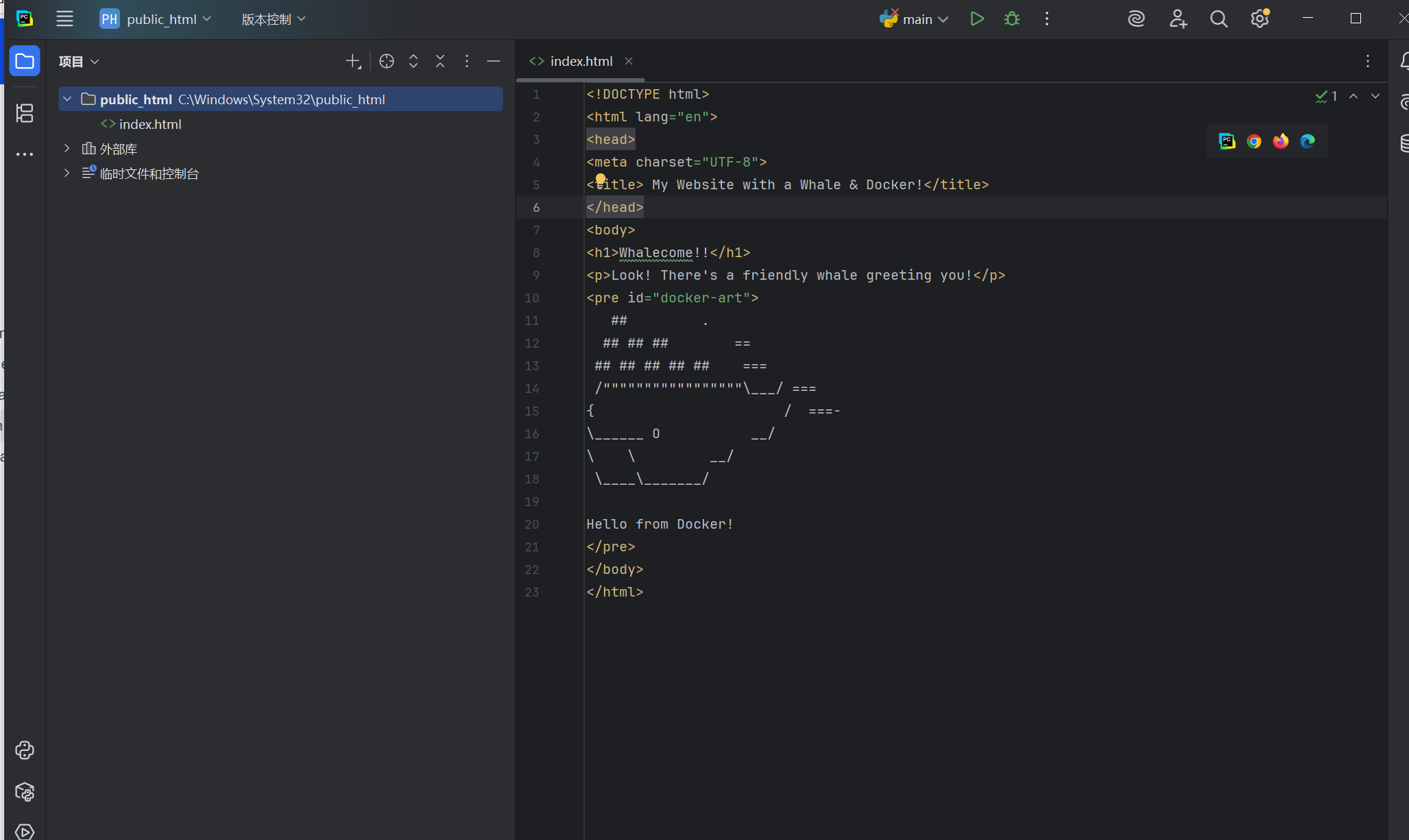
**Use a bind mount**

1. Delete the existing container by using the Docker Desktop Dashboard:
2. Create a new directory called public\_html on your host system.

mkdir public\_html

图形用户界面

AI 生成的内容可能不正确。

1. Navigate into the newly created directory public\_html and create a file called index.html with the following content. This is a basic HTML document that creates a simple webpage that welcomes you with a friendly whale.
2. It's time to run the container. The --mount and -v examples produce the same result. You can't run them both unless you remove the my\_site container after running the first one.

docker run -d --name my\_site -p 8080:80 -v .:/usr/local/apache2/htdocs/ httpd:2.4

docker run -d --name my\_site -p 8080:80 --mount type=bind,source=./,target=/usr/local/apache2/htdocs/ httpd:2.4

1. **Multi-container applications**

**Set up**

1. Get the sample application. If you have Git, you can clone the repository for the sample application. Otherwise, you can download the sample application.

Use the following command in a terminal to clone the sample application repository.

git clone https://github.com/dockersamples/nginx-node-redis

Navigate into the **nginx-node-redis** directory:

cd nginx-node-redis

Inside this directory, you'll find two sub-directories - **nginx** and **web**.

1. Download and install Docker Desktop.

**Build the images**

1. Navigate into the **nginx** directory to build the image by running the following command:

docker build -t nginx .

1. Navigate into the web directory and run the following command to build the first web image:

docker build -t web .

**Run the containers**

1. Before you can run a multi-container application, you need to create a network for them all to communicate through. You can do so using the docker network create command:

docker network create sample-app

1. Start the Redis container by running the following command, which will attach it to the previously created network and create a network alias (useful for DNS lookups):

docker run -d --name redis --network sample-app --network-alias redis redis

1. Start the first web container by running the following command:

docker run -d --name web1 -h web1 --network sample-app --network-alias web1 web

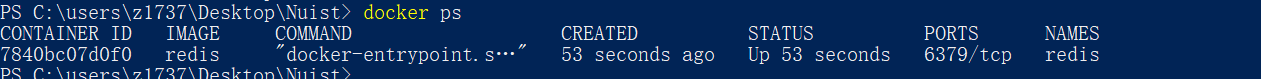
1. Start the second web container by running the following:

docker run -d --name web2 -h web2 --network sample-app --network-alias web2 web

1. Start the Nginx container by running the following command:

docker run -d --name nginx --network sample-app -p 80:80 nginx

1. Verify the containers are up by running the following command:

docker ps图形用户界面, 文本, 应用程序, 电子邮件

AI 生成的内容可能不正确。