Docker Containers for Beginners

by: Philip Mello

Follow on: Github Follow on: linkedin

Creating Docker containers is a fundamental aspect of using Docker for application development and deployment. This document provides a concise guide on how to create and manage Docker containers, including the necessary commands and best practices to ensure efficient containerization.

Abstract

This document serves as a practical guide for developers and system administrators looking to create Docker containers. It covers the essential commands and steps required to set up containers, manage their lifecycle, and highlights best practices for effective container management. By following this guide, users will gain a solid understanding of how to leverage Docker for their applications.

Getting Started with Docker

Before creating a Docker container, ensure that Docker is installed on your system. You can download and install Docker from the official Docker website.[https://www.docker.com/get-started]

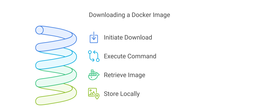
Installation command:

```
curl -fsSL https://get.docker.com -o get-docker.sh
```

Pulling an Image

To create a container, you first need a Docker image. You can pull an existing image from Docker Hub using the following command:

```
docker pull <image_name>
```



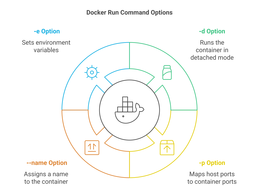
For example, to pull the latest version of the Ubuntu image, you would run:

```
docker pull ubuntu:latest
```

Creating a Container

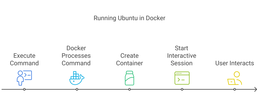
Once you have the desired image, you can create a container using the **docker run** command. The basic syntax is:

```
docker run [OPTIONS] <image_name>
```



For example, to create and run a container from the Ubuntu image, you can use:

```
docker run -it ubuntu:latest
```



Here, the **-it** option allows you to interact with the container via the terminal.

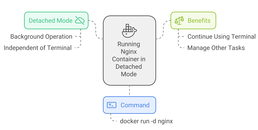
Running a Detached Container

If you want to run a container in the background (detached mode), you can use the **-d** option:

```
docker run -d <image_name>
```

For example:

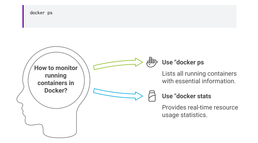
```
docker run -d nginx
```



Listing Containers

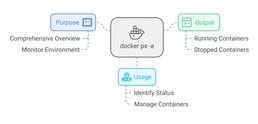
To view all running containers, use:

```
docker ps
```



To see all containers, including those that are stopped, use:

```
docker ps -a
```



Stopping and Removing Containers

To stop a running container, use the following command:

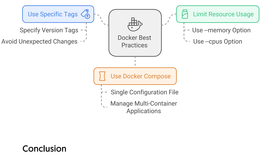
```
docker stop <container_id>
```

To remove a container, you can use:

```
docker rm <container_id>
```

Best Practices

1. **Use Specific Tags**: When pulling images, specify a version tag to avoid unexpected changes when the image is updated.
2. **Limit Resource Usage**: Use options like **--memory** and **--cpus** to limit the resources allocated to your containers.
3. **Use Docker Compose**: For multi-container applications, consider using Docker Compose to manage your containers with a single configuration file.



Conclusion

Creating Docker containers is a straightforward process that can significantly enhance your development workflow. By following the steps outlined in this document, you can efficiently create, manage, and deploy containers tailored to your application's needs. Embrace the power of Docker to streamline your development and deployment processes.