Github Desktop KASM

- Simplifying GitHub Desktop Setup with Docker

Simplifying GitHub Desktop Setup with Docker

GitHub Desktop offers a user-friendly interface for managing your Git repositories. By deploying GitHub Desktop within a Docker container, you can streamline the setup process and ensure consistency across different environments. In this article, we'll guide you through configuring GitHub Desktop using Docker Compose.

Introduction

GitHub Desktop provides a convenient way to interact with your Git repositories through a graphical interface. Dockerizing GitHub Desktop enables you to isolate its environment and dependencies, making deployment and maintenance more manageable.

Prerequisites

Before proceeding, ensure Docker is installed on your system. Refer to the official Docker documentation for installation instructions: Docker Installation Guide.

Docker Configuration

Below is a Docker Compose configuration for setting up GitHub Desktop within a Docker container:

```
services:
  github-desktop:
    image: lscr.io/linuxserver/github-desktop:latest
    container_name: github-desktop
    cap_add:
      - IPC_LOCK
    security_opt:
      - seccomp:unconfined #optional
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      - CUSTOM_USER=
```

```
      - PASSWORD=
    volumes:
      - /path/to/config:/config
    ports:
      - 3000:3000
      - 3001:3001
    shm_size: "2gb"
    restart: unless-stopped
```

Explanation

- **image**: Specifies the Docker image to use, which is the latest version of GitHub Desktop from the LinuxServer repository.
- **container_name**: Sets the name for the Docker container.
- **cap_add**: Grants additional capabilities to the container. Here, `IPC_LOCK` is added.
- **security_opt**: Sets security options for the container. In this case, `seccomp:unconfined` is specified (optional).
- **environment**: Defines environment variables for the container, including user ID (PUID), group ID (PGID), timezone (TZ), custom user, and password.
- **volumes**: Maps a local directory to the container's `/config` directory, allowing persistent storage for GitHub Desktop's configuration files.
- **ports**: Exposes ports `3000` and `3001` for GitHub Desktop's web interface.
- **shm_size**: Sets the shared memory size for the container.
- **restart**: Specifies the restart policy for the container.

Usage

1. **Create a Docker Compose file**: Copy the provided Docker Compose configuration into a file named `docker-compose.yml`.
2. **Adjust configuration**: Modify the environment variables as needed, especially the `CUSTOM_USER` and `PASSWORD` variables to match your preferences.
3. **Set up volumes**: Replace `/path/to/config` with the directory path on your host machine where you want to store GitHub Desktop's configuration files.
4. **Run GitHub Desktop**: Open a terminal, navigate to the directory containing the `docker-compose.yml` file, and run the following command:

   ```
   docker-compose up -d
   ```

   This command will download the GitHub Desktop Docker image (if not already available) and start the container in detached mode.
5. **Access GitHub Desktop**: Once the container is running, you can access GitHub Desktop's web interface by navigating to `http://localhost:3000` in your web browser.

Conclusion

By deploying GitHub Desktop with Docker, you can simplify setup and maintenance, ensuring a consistent environment for managing your Git repositories. Dockerizing GitHub Desktop provides flexibility and portability, making it easier to work with Git across different systems.