Documentations on Bookstack Docker Website Creation

Bookstack website creation detailed explanation below

- *Bookstack is a free & open-source Wiki designed for creating beautiful documentation. Featuring a simple, but powerful WYSIWYG editor it allows for teams to create detailed and useful documentation with ease.*
- *Powered by SQL and including a Markdown editor for those who prefer it, Bookstack is geared towards making documentation more of a pleasure than a chore. For more information on Bookstack visit their website and check it out: https://www.bookstackapp.com*

Supported Architecture

- *We utilize the docker manifest for multi-platform awareness. More information is available from docker here and our announcement here.*
- *Simply pulling lscr.io/linuxserver/bookstack:latest should retrieve the correct image for your arch, but you can also pull specific arch images via tags.*
- *The architectures supported by this image are:*

Architecture	Available	Tag
x86-64	✓	amd64-<version tag>
arm64	✓	arm64v8-<version tag>
armhf	✗	

Application Setup

- *The default username is admin@admin.com with the password of password, access the container at http://dockerhost:6875.*
- *This application is dependent on a MySQL database be it one you already have or a new one. If you do not already have one, set up our MariaDB container here https://hub.docker.com/r/linuxserver/mariadb/.*
- *If you intend to use this application behind a subfolder reverse proxy, such as our SWAG container or Traefik you will need to make sure that the APP_URL environment variable is set to your external domain, or it will not work.*
- *Documentation for Bookstack can be found at https://www.bookstackapp.com/docs/.*

Bookstack File & Directory Paths

- *This container ensures certain Bookstack application files & folders, such as user file upload folders, are retained within the /config folder so that they are persistent & accessible when the /config container path is bound as a volume. There may be cases, when following the Bookstack documentation, that you'll need to know how these files and folders are used relative to a non-container Bookstack installation.*
- *Below is a mapping of container /config paths to those relative within a Bookstack install directory:*

- /config container path => Bookstack relative path
- /config/www/.env => .env
- /config/www/laravel.log => storage/logs/laravel.log
- /config/www/files/ => storage/uploads/files/
- /config/www/images/ => storage/uploads/images/
- /config/www/themes/ => themes/
- /config/www/uploads/ => public/uploads/

Advanced Users (full control over the .env file)

If you wish to use the extra functionality of Bookstack such as email, Memcached, LDAP and so on you will need to make your own .env file with guidance from the Bookstack documentation.

When you create the container, do not set any arguments for any SQL settings. The container will copy an exemplary .env file to /config/www/.env on your host system for you to edit.

Usage

Here are some example snippets to help you get started creating a container.

docker-compose (recommended, click here for more info)

```
---
version: "2"
services:
  bookstack:
    image: lscr.io/linuxserver/bookstack
    container_name: bookstack
    environment:
      - PUID=1000
      - PGID=1000
      - APP_URL=https://bookstack.example.com
      - DB_HOST=bookstack_db
      - DB_PORT=3306
      - DB_USER=bookstack
      - DB_PASS=<yourdbpass>
      - DB_DATABASE=bookstackapp
    volumes:
      - ./bookstack_app_data:/config
    ports:
      - 6875:80
    restart: unless-stopped
    depends_on:
      - bookstack_db
  bookstack_db:
    image: lscr.io/linuxserver/mariadb
    container_name: bookstack_db
    environment:
      - PUID=1000
      - PGID=1000
      - MYSQL_ROOT_PASSWORD=<yourdbpass>
      - TZ=Europe/London
      - MYSQL_DATABASE=bookstackapp
      - MYSQL_USER=bookstack
      - MYSQL_PASSWORD=<yourdbpass>
    volumes:
      - ./bookstack_db_data:/config
    restart: unless-stopped
```

docker cli (click here for more info)

```
docker run -d \
  --name=bookstack \
```

```
-e PUID=1000 \
-e PGID=1000 \
-e TZ=Etc/UTC \
-e APP_URL=<yourbaseurl> \
-e DB_HOST=<yourdbhost> \
-e DB_PORT=<yourdbport> \
-e DB_USER=<yourdbuser> \
-e DB_PASS=<yourdbpass> \
-e DB_DATABASE=bookstackapp \
-p 6875:80 \
-v /path/to/data:/config \
--restart unless-stopped \
lscr.io/linuxserver/bookstack:latest
```

Parameters

Container images are configured using parameters passed at runtime (such as those above). These parameters are separated by a colon and indicate <external>:<internal> respectively. For example, -p 8080:80 would expose port 80 from inside the container to be accessible from the host's IP on port 8080 outside the container.

Parameter	Function
-p 80	**will map the container's port 80 to port 6875 on the host**
-e PUID=1000	**for UserID - see below for explanation**
-e PGID=1000	**for GroupID - see below for explanation**
-e TZ=Etc/UTC	**specify a timezone to use, see this list.**
-e APP_URL=<yourbaseurl>	**for specifying the IP:port or URL your application will be accessed on (ie. http://192.168.1.1:6875 or https://bookstack.mydomain.com**
-e DB_HOST=<yourdbhost>	**for specifying the database host**
-e DB_PORT=<yourdbport>	**for specifying the database port if not default 3306**
-e DB_USER=<yourdbuser>	**for specifying the database user**

-e DB_PASS=<yourdbpass>	**for specifying the database password (non-alphanumeric passwords must be properly escaped.)**
-e DB_DATABASE=bookstackapp	**for specifying the database to be used**
-v /config	**this will store any uploaded data on the docker host**

Environment variables from files (Docker secrets)

You can set any environment variable from a file by using a special prepend FILE__.

As an example:

-e FILE__PASSWORD=/run/secrets/mysecretpassword

Will set the environment variable PASSWORD based on the contents of the /run/secrets/mysecretpassword file.

Umask for running applications

For all of our images we provide the ability to override the default umask settings for services started within the containers using the optional -e UMASK=022 setting. Keep in mind umask is not chmod it subtracts from permissions based on its value it does not add. Please read up here before asking for support.

User / Group Identifiers

When using volumes (-v flags) permissions issues can arise between the host OS and the container, we avoid this issue by allowing you to specify the user PUID and group PGID.

Ensure any volume directories on the host are owned by the same user you specify and any permissions issues will vanish like magic.

In this instance PUID=1000 and PGID=1000, to find yours use id user as below:

Updating Info

Most of our images are static, versioned, and require an image update and container recreation to update the app inside. With some exceptions (ie. nextcloud, plex), we do not recommend or support updating apps inside the container. Please consult the Application Setup section above to see if it is recommended for the image.

Below are the instructions for updating containers:

Via Docker Compose

- *Update all images: docker-compose pull*
    - *or update a single image: docker-compose pull bookstack*
- *Let compose update all containers as necessary: docker-compose up -d*
    - *or update a single container: docker-compose up -d bookstack*
- *You can also remove the old dangling images: docker image prune*

Via Docker Run

- *Update the image: docker pull lscr.io/linuxserver/bookstack:latest*
- *Stop the running container: docker stop bookstack*
- *Delete the container: docker rm bookstack*
- *Recreate a new container with the same docker run parameters as instructed above (if mapped correctly to a host folder, your /config folder and settings will be preserved)*
- *You can also remove the old dangling images: docker image prune*

Via Watchtower auto-updater (only use if you don't remember the original parameters)

- *Pull the latest image at its tag and replace it with the same env variables in one run:*
- *docker run --rm *
  *-v /var/run/docker.sock:/var/run/docker.sock *
  *containrrr/watchtower *
  --run-once bookstack

- *You can also remove the old dangling images: docker image prune*
- *Lastly by uploading the cheat sheets to the booksheat such as Git, Linux, Docker, Kubernetes*