

Example 3:

Setting Environment Variable:

Environment variables are **key-value pairs** used to configure applications.

To set environment variables we use “-e”

```
docker run -it -e MY_NAME=sakura_winter --name env_var ubuntu bash
```

```
sumitkumarmehta@sumits-MacBook-Air ~ % docker run -it -e NAME=SAKURA_WINTER --name env_var ubuntu bash
```



env_var

f9cd80beda63

[ubuntu](#)

To check the variable go to the bash inside container and type:

```
echo $NAME
```

```
root@f9cd80beda63:/# echo $NAME
SAKURA_WINTER
```

NOTE:

The **-it** flag enables running the Docker container in interactive mode.

To exit interactive mode, type **exit**.

Multiple Environment Variable:

```
docker run -e APP_ENV=production -e APP_VERSION=1.0
nginx
```

```
sumitkumarmehta@sumits-MacBook-Air ~ % sumitkumarmehta@sumits-MacBook-Air ~ % docker run -e APP_ENV=production -e APP_VERSION=1.0
nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2026/02/26 01:44:42 [notice] 1#1: using the "epoll" event method
2026/02/26 01:44:42 [notice] 1#1: nginx/1.29.4
2026/02/26 01:44:42 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2026/02/26 01:44:42 [notice] 1#1: OS: Linux 6.12.68-linuxkit
2026/02/26 01:44:42 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2026/02/26 01:44:42 [notice] 1#1: start worker processes
2026/02/26 01:44:42 [notice] 1#1: start worker process 29
2026/02/26 01:44:42 [notice] 1#1: start worker process 30
2026/02/26 01:44:42 [notice] 1#1: start worker process 31
2026/02/26 01:44:42 [notice] 1#1: start worker process 32
2026/02/26 01:44:42 [notice] 1#1: start worker process 33
2026/02/26 01:44:42 [notice] 1#1: start worker process 34
2026/02/26 01:44:42 [notice] 1#1: start worker process 35
2026/02/26 01:44:42 [notice] 1#1: start worker process 36
2026/02/26 01:44:42 [notice] 1#1: start worker process 37
2026/02/26 01:44:42 [notice] 1#1: start worker process 38
```

Another Example with MYSQL Container

```
docker run -d \
-e MYSQL_ROOT_PASSWORD=root123 \
-e MYSQL_DATABASE=college \
-e MYSQL_USER=admin \
-e MYSQL_PASSWORD=admin123 \
mysql:8
```

NOTE: Without these variables, MySQL container will fail to start

Passing Environment Variable from host system:

Step 1: In your host machine terminal type:

```
export APP_PORT=8080
Echo $APP_PORT
```

 **sumitkumarmehta — -zsh — 80x24**

```
Last login: Thu Feb 26 07:05:50 on console
[sumitkumarmehta@sumits-MacBook-Air ~ % export APP_PORT=8080
[sumitkumarmehta@sumits-MacBook-Air ~ % echo $APP_PORT
8080
```

Step 2: run docker command using:

```
docker run -e APP_PORT nginx env
```

env → this command prints all the environment variable

```
sumitkumarmehta@sumits-MacBook-Air ~ % export APP_PORT=8080
sumitkumarmehta@sumits-MacBook-Air ~ % echo $APP_PORT
8080
sumitkumarmehta@sumits-MacBook-Air ~ % docker run -e APP_PORT nginx env
HOSTNAME=a23603739fc5
HOME=/root
APP_PORT=8080
PKG_RELEASE=1~trixie
DYNPKG_RELEASE=1~trixie
ACME_VERSION=0.3.1
NGINX_VERSION=1.29.4
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
NJS_VERSION=0.9.4
NJS_RELEASE=1~trixie
PWD=/
```