Install in a Container

Overview

Yellowfin is available as a variety of images for Docker and Kubernetes. Both these platforms use container technology to deliver the software. You can choose between two types of Yellowfin containers—the Yellowfin All-In-One image (software and database), and the Yellowfin App-Only image (software only).

Yellowfin image types

Yellowfin All-In-One image

The Yellowfin All-In-One image contains both the Yellowfin application as well as the Yellowfin repository database, which in this case is an embedded PostgreSQL database.

This image will not persist data outside of the Docker container, and all content will be lost when the container is destroyed. With this in mind, we recommend the following uses cases.

Use cases

This image is most suited to the following use cases:

- Short-term POCs and demos.
- Testing Yellowfin functionality in a sandboxed environment.

Avoid using this image for the following use cases:

- Production deployments (all content is lost when the container is destroyed).
- Clustered environments (this image doesn't support this functionality).

Yellowfin All-In-One image configuration options

The Yellowfin All-In-One image has the following environment variable available for configuration:

Configuration Item	Description	Example
Application Memory	Specify the number of megabytes of memory to be assigned to the Yellowfin application. If unset, Yellowfin will use the Java default (usually 25% of system RAM)	-e APP_MEMORY =4096

Yellowfin App-Only image

The Yellowfin App Only image contains only the Yellowfin application. It must therefore be connected to an existing repository database.

You can use this image to deploy a single instance, discrete instances or a Yellowfin Cluster. This image is suitable for both production and non-production environments, as data persists on the external Yellowfin repository, so Yellowfin data will be retained, even if containers are destroyed.

Use cases

This image is most suited to the following use cases:

- Long-term instances of Yellowfin, where data persistence is important.
- Clustered Yellowfin deployments.

Yellowfin App-Only image configuration options

The Yellowfin App-Only image has the following environment variables available for configuration:

Configuration Item	Description	Example
JDBC Driver Name, JDBC_CLASS_NAME	Configure the JDBC Driver Class for connecting to the Yellowfin Repository (Required)	-e JDBC_CLASS_NAME=org.postgresql.Driver
Repository URL, JDBC_CONN_URL	Specify the Connection URL to the Repository Database (Required)	-e JDBC_CONN_URL=jdbc:postgresql://host:5432/yf
Repository Username, JDBC_CONN_USER	Specify the Database User required to access the Repository Database (Required)	-e JDBC_CONN_USER=dba
Repository Password, JDBC_CONN_PASS	Specify the Database Password required to access the Repository Database. This can be encrypted. (Required)	-e JDBC_CONN_PASS=secret
Application Memory, APP_MEMORY	Specify the number of megabytes of memory to be assigned to the Yellowfin application. If unset, Yellowfin will use the Java default (usually 25% of System RAM)	-e APP_MEMORY=4096
DB Password Encrypted, JDBC_CONN_ENCRYPTED	Specify whether the Database Password is encrypted (true/false)	-e JDBC_CONN_ENCRYPTED=true
Connection Pool Size, JDBC_MAX_COUNT	Specify the maximum size of the Repository Database connection pool. (Default: 25)	-e JDBC_MAX_COUNT=25
Default Welcome Page, WELCOME_PAGE	Specify the default index page.	-e WELCOME_PAGE=custom_index.jsp
Internal Application HTTP Port	Specify the internal HTTP port. (Default: 8080)	-e APP_SERVER_PORT=9090
Internal Shutdown Port	Specify the internal shutdown port. (Default: 8083)	-e TCP_PORT=9093
Proxy Port, PROXY_PORT	External Proxy Port	-e PROXY_PORT=443
Proxy Scheme, PROXY_SCHEME	External Proxy Scheme (http/https)	-e PROXY_SCHEME=https
Proxy Host, PROXY_HOST	External Proxy Host or IP address	-e PROXY_HOST=reporting.company.com
External Cluster Address, CLUSTER_ADDRESS	External Cluster Address for Cluster Messaging. Usually the host or IP address of the Docker Host, or for Docker Swarm and Kubernetes, the DNS name of the container.	-e CLUSTER_ADDRESS=10.10.10.23
External Cluster Port, CLUSTER_PORT	A Unique TCP port for this container to receive Cluster Messages from other nodes	-e CLUSTER_ADDRESS=7801
Internal Cluster Network Adapter, CLUSTER_INTERFACE	Specify the docker interface to bind Cluster Messages to. Defaults to eth0, but this may need to be changed for Kubernetes and Docker Swarm	-e CLUSTER_INTERFACE=match-interface:eth1
Background Processing Task Types, NODE_BACKGROUND_TASKS	Comma separated list of which background Task Types can be run on this node. NODE_PARALLEL_TASKS must also be updated if this item is specified. If unspecified, all Task Types will be enabled.	-e NODE_BACKGROUND_TASKS= FILTER_CACHE, ETL_PROCESS_TASK
Background Task Processing Jobs, NODE_PARALLEL_TASKS	Comma separated list of the number of concurrent tasks for each Task Type that can be run on this node. The number of elements passed here must match the number of Task Types passed by NODE_BACKGROUND_TASKS	-e NODE_PARALLEL_TASKS=5,4
Additional Libraries URL, LIBRARY_ZIP	URL to a Zip file that contains additional libraries to be extracted into lib folder of Yellowfin. This can be used to add additional JDBC drivers or custom plugins to Yellowfin. Make sure that the path is not included with zip entries in the archive.	-e LIBRARY_ZIP=http://lib-host/libraries.zip

top

Yellowfin image downloads and builds

Yellowfin is available for download in a variety of formats.

The core Yellowfin installer app is available from the Yellowfin portal.

Yellowfin Docker images are available on Docker Hub.

Yellowfin assets (where you can build your own Docker images) are available on GitHub.

You will need a license to activate Yellowfin. If you don't already have a license, we can provide evaluation licenses or full licenses. Please get in touch to request a license.

Yellowfin on GitHub

The Yellowfin repository on Github provides the two Yellowfin Docker images as downloadable Dockerfiles, as well as a copy of the deployment file examples on these pages.

To build a Docker image, follow these steps (and replace My_Docker_Registry with your Docker registry name). The steps below use the latest build as the example.

1. Move the downloaded Yellowfin Dockerfile to a suitable directory on your system
2. Download the Yellowfin jar file from https://portal.yellowfinbi.com/yf_latestbuild.jsp
3. Open a terminal session and navigate to the folder containing the Yellowfin Dockerfile and the Yellowfin jar file.
4. Build the Docker image from the Dockerfile using the terminal command:
   docker build -t yellowfin-app-only
5. Tag the image with a version using the terminal command:
   docker tag yellowfin-app-only:latest My_Docker_Registry/yellowfin-app-only:latest
6. Push the image to a Docker registry
   docker push My_Docker_Registry/yellowfin-app-only:latest

Yellowfin on Docker Hub

If you prefer to pull the prebuilt Docker images directly, the Yellowfin Docker Hub account account hosts both the Yellowfin App-Only Image Repository and Yellowfin All-In-One Image Repository. Both are versioned to match official Yellowfin release builds.

You may also see some interim builds, but these are exceptional builds for specific partners that are untested, and they should therefore not be used.

To pull an image, use a combination of command and suffix, outlined in the following table.

Image type	Command	Possible suffixes
All-In-One image	`docker pull yellowfinbi/yellowfin-all-in-one:`	`latest` `x.x.x` (eg, `9.5.1` for Yellowfin 9.5.1 or `9.6.0` for Yellowfin 9.6)
App-Only image	`docker pull yellowfinbi/yellowfin-app-only:`	`latest` (use with caution, see note below) `x.x.x` (eg, `9.5.1` for Yellowfin 9.5.1 or `9.6.0` for Yellowfin 9.6)

For example, to pull the latest Yellowfin All-In-One Docker image:

```
docker pull yellowfinbi/yellowfin-all-in-one:latest
```

Or to pull a particular version of the Yellowfin App-Only Docker image:

```
docker pull yellowfinbi/yellowfin-app-only:9.5.1
```

To use a specific version of Yellowfin, search for it in the "Tags" section of the image repository before pulling it.

⚠ **We highly recommend you use a specific version number rather than 'latest' when pulling the App-Only image to avoid accidental updates during start-up.**

top

Deployment examples

The steps we've written for each type of containerized deployment should work for all major cloud providers, as well as on-premises environments.

We wrote and tested these steps on Amazon Web Services using the following services:

- EC2 instances for the Docker and Docker Swarm environments
- AWS EKS for the Kubernetes environment
- AWS ECR as a Docker registry (we used it for storing the Yellowfin Docker images for all our environments)

Although unlikely, modifications to the deployment files may be required on other cloud environments.

top

Section navigation

Current topic - Install in a Container

The page is part of the Install in a Container topic contains the following pages, split by Docker and Kubernetes:

This page is part of the Install And Deploy Yellowfin section of the wiki, which has these topics:

Install on Premises

Install in the Cloud

- Yellowfin for AWS
- Yellowfin for Azure
- Yellowfin for Google Cloud Platform

Install in a container

Install in a Container

- Docker
- Kubernetes
- Upgrading Yellowfin Container Deployment

Deploy Yellowfin

Deploy Yellowfin

- Logs and Logging
- Yellowfin Directory Structure
- User Welcome

Advanced Deployments

Advanced Deployments

- Clustering Guide
- Yellowfin Server Specification
- Automate Yellowfin Deployment on Linux
- SAML Bridge
- Standalone Configuration Tools

top