Install New Relic Java agent for Docker

On this page

On this page

This document explains a basic installation of the APM agent for Java applications in a Docker container. We discuss required configurations and also explore some optional configurations, including:

- How to use identical New Relic configuration files for each container, regardless of the environment where the containers are used
- How to use the Docker layer when every agent in every environment needs slightly different configuration data
- How to disable the New Relic agent in some environments and enable it in others

Although we don't discuss advanced options here, you can install the Java agent in Docker volumes and use your Docker container image in other software such as Swarm, ECS, AKS, EKS, OpenShift, and Kubernetes. Our Docker examples refer to Tomcat, so if you are using another application server, refer to your vendor's documentation.

To use Java or any other agent, as well as the rest of our observability platform, join the New Relic family! Sign up to create your free account in only a few seconds. Then ingest up to 100GB of data for free each month. Forever.

Get the Java agent

Download `newrelic-java.zip` using `curl`, `Invoke-WebRequest` (PowerShell), or the New Relic UI:

Hide All

Download using `curl`

Complete the following:

1. Start a command-line session.
2. Change to a temporary directory where you can download the zip file.
3. Execute this `curl` command:

```
curl -O https://download.newrelic.com/newrelic/java-agent/newrelic-agent/current/newrelic-java.zip
```

4. Unzip `newrelic-java.zip`

Download using `Invoke-WebRequest` (PowerShell)

Complete the following:

1. Start a PowerShell session.
2. Change to a temporary directory where you can download the zip file.
3. Execute this PowerShell command:

```
Invoke-WebRequest -Uri https://download.newrelic.com/newrelic/java-agent/newrelic-agent/current/newrelic-java.zip -OutFile newrelic-java.zip
```

4. Unzip `newrelic-java.zip`:

```
Expand-Archive -Path newrelic-java.zip -DestinationPath DESTINATION_PATH
```

Download from the New Relic UI

Complete the following:

1. Log in to New Relic.
2. From the account dropdown in the New Relic UI, select **Account settings**.
3. In the right sidebar under **Most recent**, select the Java agent, and save the `newrelic-java.zip` to a temporary directory.
4. Unzip `newrelic-java.zip`.

Set up the installation directory

You can unzip the `newrelic-java.zip` file wherever it is convenient for you. In the subsequent sections we assume you extracted it in the current working directory, which puts the files we need in `./newrelic`.

Modify startup scripts

The startup script that contains the command to start your application server must include Java's built-in argument `-javaagent`. We recommend that you set this argument with the `JAVA_OPTS` environment variable. The value of that argument must contain the location where you `ADD` the Java APM agent's jar file to the image.

For example, with Tomcat, use commands like these in the `Dockerfile`:

```
RUN mkdir -p /usr/local/tomcat/newrelic
ADD ./newrelic/newrelic.jar /usr/local/tomcat/newrelic/newrelic.jar
ENV JAVA_OPTS="$JAVA_OPTS -javaagent:/usr/local/tomcat/newrelic/newrelic.jar"
```

Set agent configurations

By default, agent behavior is controlled by configuration entries in `newrelic.yml`, which is typically located in the same directory as the agent. This section explains how to override these `newrelic.yml` configurations by using environment variables or Java system properties in the `Dockerfile`.

Before we look at some specific configurations, here's how to load `newrelic.yml` using the `Dockerfile`:

```
ADD ./newrelic/newrelic.yml /usr/local/tomcat/newrelic/newrelic.yml
```

For a basic Docker installation, complete these configurations:

- Application name
- License key
- Logs
- Environment (optional)
- Enabled (optional)

Application name

The application name is a configuration you set to identify your application in New Relic.

> You can reuse an application name for multiple apps serving the same role so that all the data from those apps rolls up into the same logical application in New Relic. For more detail about additional grouping options, see Use multiple names for an app.

Replace *MY_APP_NAME* with your application name in one of these `Dockerfile` commands:

Option	Command
Environment variable	`ENV NEW_RELIC_APP_NAME="MY_APP_NAME"`

Option	Command
Java system property	`ENV JAVA_OPTS="$JAVA_OPTS -Dnewrelic.config.app_name='MY_APP_NAME'"`

After you boot the container, your application name appears in New Relic.

License key

This configuration is required for you to report on any data in your New Relic Account.

To copy your license key:

1. Go to **one.newrelic.com** > **(account dropdown)** > **Account settings**.
2. Under **Account information**, copy the license key.
3. In one of these `Dockerfile` commands, replace *MY_LICENSE_KEY* with your license key:

Option	Command
Environment variable	`ENV NEW_RELIC_LICENSE_KEY="MY_LICENSE_KEY"`
Java system property	`ENV JAVA_OPTS="$JAVA_OPTS -Dnewrelic.config.license_key='MY_LICENS`

Logs

By default, logs are written into the logs directory relative to the location of `newrelic.jar`. Make sure that the user account that starts your application server also has the right to perform tasks such as:

- Creating the logs directory.
- Creating and appending to the log files in that directory.

Here's a Dockerfile example where `tomcat` is the user who starts Tomcat:

```
RUN mkdir -p /usr/local/tomcat/newrelic/logs
RUN chown -R tomcat:tomcat /usr/local/tomcat/newrelic/logs
```

You can also send the logs to `STDOUT` by adding one of the following to the Dockerfile:

Option	Command
Environment Variable	`ENV NEW_RELIC_LOG_FILE_NAME=STDOUT`

Option	Command
Java system property	`ENV JAVA_OPTS=-Dnewrelic.config.log_file_name=STDOUT`

Environment (optional)

You can pass either a Java property or an environment variable to determine which of the environment-specific stanzas the agent uses in `newrelic.yml`. Use this approach if you prefer to have the `newrelic.yml` file control environment-specific configurations instead of passing all the configurations via Docker.

Here's a `Dockerfile` example of passing the `newrelic.environment` Java system property via Docker to use the custom value `dev` in the environment stanza of `newrelic.yml`:

1. Using the shell form of the CMD instruction, include a reference to a new environment variable you choose (for example, `ENV`):

   ```
   CMD java -Dnewrelic.environment=$ENV -jar myjar.jar
   ```

2. In your `docker run` command line, include an argument to set the environment variable in the container:

   ```
   docker run -it -e "ENV=dev" myDockerImage
   ```

> If you don't specify a value for `newrelic.environment`, the agent assumes it is running in your production environment and uses the values from the main body of the configuration file.

Enabled (optional)

This configuration controls whether the agent is enabled. Let's say you want the same Docker image for every installation. However, you don't want to run the New Relic agent every time an engineer spins up a test app because you don't want to run up your instance count.

This problem can be solved using the `newrelic.environment` Java system property.

1. In the main body of `newrelic.yml`, disable the Java agent by setting `enabled: false`.
2. In specific environment stanzas of `newrelic.yml`, set `enabled: true`.

Then, you can run specific agents by specifying the environment at runtime.

Additional Tomcat Dockerfile examples

Hide All

Tomcat with environment and Java system properties

```
FROM tomcat:9
# Add the newrelic.jar and -javaagent parameters
RUN mkdir -p /usr/local/tomcat/newrelic
ADD ./newrelic/newrelic.jar /usr/local/tomcat/newrelic/
ENV JAVA_OPTS="$JAVA_OPTS -javaagent:/usr/local/tomcat/newrelic/newrelic.jar"
# Add the configuration file
ADD ./newrelic/newrelic.yml /usr/local/tomcat/newrelic/
# An example of setting a system property config
ENV JAVA_OPTS="$JAVA_OPTS -Dnewrelic.config.app_name='My Application'"
# An example of setting an Environment variable config
ENV NEW_RELIC_LICENSE_KEY="license_key"
# Config to include the agent logs in Docker's stdout logging
ENV JAVA_OPTS="$JAVA_OPTS -Dnewrelic.config.log_file_name=STDOUT"
EXPOSE 8080
CMD ["catalina.sh", "run"]
```

How to start an application with the Java agent

```
FROM openjdk:8
ADD my-application.jar /app
ADD newrelic.jar  /app
ADD newrelic.yml  /app
ENV NEW_RELIC_APP_NAME="My Application"
ENV NEW_RELIC_LICENSE_KEY="license_key"
ENV NEW_RELIC_LOG_FILE_NAME="STDOUT"
ENTRYPOINT ["java","-javaagent:/app/newrelic.jar","-jar","/app/my-application.jar"]
```

Next steps

Now that you have a basic agent installation in Docker, here are some additional steps to consider:

- Review other configurations for the agent.
- Read a detailed Explorers Hub post about Docker and New Relic.

For more help

If you need more help, check out these support and learning resources:

- Browse the Explorers Hub to get help from the community and join in discussions.
- Find answers on our sites and learn how to use our support portal.
- Run New Relic Diagnostics, our troubleshooting tool for Linux, Windows, and macOS.
- Review New Relic's data security and licenses documentation.