

# Creating a container image for use on Amazon ECS

[PDF \(ecs-dg.pdf#create-container-image\)](#)[RSS \(amazon-ecs-release-notes.rss\)](#)

Amazon ECS uses Docker images in task definitions to launch containers. Docker is a technology that provides the tools for you to build, run, test, and deploy distributed applications in containers. Docker provides a walkthrough on deploying containers on Amazon ECS. For more information, see [Deploying Docker containers on Amazon ECS](#) [\(https://docs.docker.com/engine/context/ecs-integration/\)](https://docs.docker.com/engine/context/ecs-integration/).

The purpose of the steps outlined here is to walk you through creating your first Docker image and pushing that image to Amazon ECR, which is a container registry, for use in your Amazon ECS task definitions. This walkthrough assumes that you possess a basic understanding of what Docker is and how it works. For more information about Docker, see [What is Docker?](#) [\(http://aws.amazon.com/docker/\)](http://aws.amazon.com/docker/) and the [Docker overview](#) [\(https://docs.docker.com/engine/docker-overview/\)](https://docs.docker.com/engine/docker-overview/).

## Important

AWS and Docker have collaborated to make a simplified developer experience that enables you to deploy and manage containers on Amazon ECS directly using Docker tools. You can now build and test your containers locally using Docker Desktop and Docker Compose, and then deploy them to Amazon ECS on Fargate. To get started with the Amazon ECS and Docker integration, download Docker Desktop and optionally sign up for a Docker ID. For more information, see [Docker Desktop](#)  [\(https://www.docker.com/products/docker-desktop\)](https://www.docker.com/products/docker-desktop) and [Docker ID signup](#)  [\(https://hub.docker.com/signup/awssedge?utm\\_source=awssedge\)](https://hub.docker.com/signup/awssedge?utm_source=awssedge).

## Prerequisites

Before you begin, ensure the following prerequisites are met.

- Ensure you have completed the Amazon ECR setup steps. For more information, see [Setting up for Amazon ECR](#)  [\(https://docs.aws.amazon.com/AmazonECR/latest/userguide/get-set-up-for-amazon-ecr.html\)](https://docs.aws.amazon.com/AmazonECR/latest/userguide/get-set-up-for-amazon-ecr.html) in the *Amazon Elastic Container Registry User Guide*.

- Your user has the required IAM permissions to access and use the Amazon ECR service. For more information, see [Amazon ECR managed policies](https://docs.aws.amazon.com/AmazonECR/latest/userguide/security-iam-awsmanpol.html) (<https://docs.aws.amazon.com/AmazonECR/latest/userguide/security-iam-awsmanpol.html>) .
- You have Docker installed. For Docker installation steps for Amazon Linux 2, see [Installing Docker on Amazon Linux 2 \(#create-container-image-install-docker\)](#) . For all other operating systems, see the Docker documentation at [Docker Desktop overview](#) <https://docs.docker.com/desktop/> .
- You have the AWS CLI installed and configured. For more information, see [Installing the AWS Command Line Interface](#) (<https://docs.aws.amazon.com/cli/latest/userguide/installing.html>) in the *AWS Command Line Interface User Guide*.

If you don't have or need a local development environment and you prefer to use an Amazon EC2 instance to use Docker, we provide the following steps to launch an Amazon EC2 instance using Amazon Linux 2 and install Docker Engine and the Docker CLI.

## ► Installing Docker on Amazon Linux 2

Docker Desktop is an easy-to-install application for your Mac or Windows environment that enables you to build and share containerized applications and microservices. Docker Desktop includes Docker Engine, the Docker CLI client, Docker Compose, and other tools that are helpful when using Docker with Amazon ECS. For more information about how to install Docker Desktop on your preferred operating system, see [Docker Desktop overview](#) <https://docs.docker.com/desktop/> .

### To install Docker on an Amazon EC2 instance

1. Launch an instance with the Amazon Linux 2 AMI. For more information, see [Launching an instance](#) (<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/launching-instance.html>) in the *Amazon EC2 User Guide for Linux Instances*.
2. Connect to your instance. For more information, see [Connect to your Linux instance](#) (<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html>) in the *Amazon EC2 User Guide for Linux Instances*.
3. Update the installed packages and package cache on your instance.

```
sudo yum update -y
```

4. Install the most recent Docker Engine package.

```
sudo amazon-linux-extras install docker
```

### Important

This step assumes you are using the Amazon Linux 2 AMI for your instance. For all other operating systems, see [Docker Desktop overview](https://docs.docker.com/desktop/) [\(https://docs.docker.com/desktop/\)](https://docs.docker.com/desktop/) .

5. Start the Docker service.

```
sudo service docker start
```

(Optional) To ensure that the Docker daemon starts after each system reboot, run the following command:

```
sudo systemctl enable docker
```

6. Add the `ec2-user` to the `docker` group so you can execute Docker commands without using `sudo` .

```
sudo usermod -a -G docker ec2-user
```

7. Log out and log back in again to pick up the new `docker` group permissions. You can accomplish this by closing your current SSH terminal window and reconnecting to your instance in a new one. Your new SSH session will have the appropriate `docker` group permissions.
8. Verify that you can run Docker commands without `sudo` .

```
docker info
```

### Note

In some cases, you may need to reboot your instance to provide permissions for the `ec2-user` to access the Docker daemon. Try rebooting your instance if you see the following error:

```
Cannot connect to the Docker daemon. Is the  
docker daemon running on this host?
```

## Create a Docker image

Amazon ECS task definitions use Docker images to launch containers on the container instances in your clusters. In this section, you create a Docker image of a simple web application, and test it on your local system or Amazon EC2 instance, and then push the image to the Amazon ECR container registry so you can use it in an Amazon ECS task definition.

### To create a Docker image of a simple web application

1. Create a file called `Dockerfile`. A Dockerfile is a manifest that describes the base image to use for your Docker image and what you want installed and running on it. For more information about Dockerfiles, go to the [Dockerfile Reference](https://docs.docker.com/engine/reference/builder/) (<https://docs.docker.com/engine/reference/builder/>).

```
touch Dockerfile
```

2. Edit the `Dockerfile` you just created and add the following content.

```
FROM ubuntu:18.04

# Install dependencies
RUN apt-get update && \
    apt-get -y install apache2

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo '. /etc/apache2/envvars' > /root/run_apache.sh && \
    echo 'mkdir -p /var/run/apache2' >> /root/run_apache.sh && \
    echo 'mkdir -p /var/lock/apache2' >> /root/run_apache.sh && \
    echo '/usr/sbin/apache2 -D FOREGROUND' >> \
    /root/run_apache.sh && \
    chmod 755 /root/run_apache.sh
```

```
EXPOSE 80
```

```
CMD /root/run_apache.sh
```

This Dockerfile uses the Ubuntu 18.04 image. The RUN instructions update the package caches, install some software packages for the web server, and then write the "Hello World!" content to the web server's document root. The EXPOSE instruction exposes port 80 on the container, and the CMD instruction starts the web server.

3. Build the Docker image from your Dockerfile.

#### Note

Some versions of Docker may require the full path to your Dockerfile in the following command, instead of the relative path shown below.


```
docker build -t hello-world .
```

4. Run **docker images** to verify that the image was created correctly.

```
docker images --filter reference=hello-world
```

Output:

| REPOSITORY  | TAG    | IMAGE ID     |   |
|-------------|--------|--------------|---|
| hello-world | latest | e9ffedc8c286 | 4 |
| minutes ago | 241MB  |              |   |

5. Run the newly built image. The `-p 80:80` option maps the exposed port 80 on the container to port 80 on the host system. For more information about **docker run**, go to the [Docker run reference](https://docs.docker.com/engine/reference/run/)  (<https://docs.docker.com/engine/reference/run/>) .

```
docker run -t -i -p 80:80 hello-world
```

#### Note

Output from the Apache web server is displayed in the terminal window. You can ignore the "Could not reliably determine the server's fully qualified domain name" message.

6. Open a browser and point to the server that is running Docker and hosting your container.
  - If you are using an EC2 instance, this is the **Public DNS** value for the server, which is the same address you use to connect to the instance with SSH. Make sure that the security group for your instance allows inbound traffic on port 80.
  - If you are running Docker locally, point your browser to <http://localhost/> [↗](#) (<http://localhost/>) .
  - If you are using **docker-machine** on a Windows or Mac computer, find the IP address of the VirtualBox VM that is hosting Docker with the **docker-machine ip** command, substituting *machine-name* with the name of the docker machine you are using.

```
docker-machine ip machine-name
```

You should see a web page with your "Hello World!" statement.

7. Stop the Docker container by typing **Ctrl + c**.

## Push your image to Amazon Elastic Container Registry

Amazon ECR is a managed AWS Docker registry service. You can use the Docker CLI to push, pull, and manage images in your Amazon ECR repositories. For Amazon ECR product details, featured customer case studies, and FAQs, see the [Amazon Elastic Container Registry product detail pages](http://aws.amazon.com/ecr) [↗](#) (<http://aws.amazon.com/ecr>) .

### To tag your image and push it to Amazon ECR

1. Create an Amazon ECR repository to store your hello-world image. Note the `repositoryUri` in the output.

Substitute `region`, with your AWS Region, for example, `us-east-1`.

```
aws ecr create-repository --repository-name hello-repository  
--region region
```

Output:

```
{  
  "repository": {  
    "registryId": "aws_account_id",  
    "repositoryName": "hello-repository",  
    "repositoryArn":
```

```
"arn:aws:ecr:region:aws_account_id:repository/hello-repository",
    "createdAt": 1505337806.0,
    "repositoryUri":
    "aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository"
  }
}
```

2. Tag the hello-world image with the `repositoryUri` value from the previous step.

```
docker tag hello-world
aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

3. Run the **aws ecr get-login-password** command. Specify the registry URI you want to authenticate to. For more information, see [Registry Authentication](https://docs.aws.amazon.com/AmazonECR/latest/userguide/Registries.html#registry_auth) ([https://docs.aws.amazon.com/AmazonECR/latest/userguide/Registries.html#registry\\_auth](https://docs.aws.amazon.com/AmazonECR/latest/userguide/Registries.html#registry_auth)) in the *Amazon Elastic Container Registry User Guide*.

```
aws ecr get-login-password | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Output:

```
Login Succeeded
```

### Important

If you receive an error, install or upgrade to the latest version of the AWS CLI. For more information, see [Installing the AWS Command Line Interface](https://docs.aws.amazon.com/cli/latest/userguide/installing.html) (<https://docs.aws.amazon.com/cli/latest/userguide/installing.html>) in the *AWS Command Line Interface User Guide*.

4. Push the image to Amazon ECR with the `repositoryUri` value from the earlier step.

```
docker push
aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

---

## Clean up

To continue on with creating an Amazon ECS task definition and launching a task with your container image, skip to the [Next steps \(#create-container-image-next-steps\)](#) . When you are done experimenting with your Amazon ECR image, you can delete the repository so you are not charged for image storage.

```
aws ecr delete-repository --repository-name hello-repository --  
region region --force
```

---

## Next steps

After you have created and pushed your container image to Amazon ECR, you should consider the following next steps.

- [Getting started with Amazon ECS using the classic console \(./getting-started-console.html\)](#)
- [Tutorial: Creating a cluster with a Fargate Linux task using the AWS CLI \(./ECS\\_AWSCLI\\_Fargate.html\)](#)