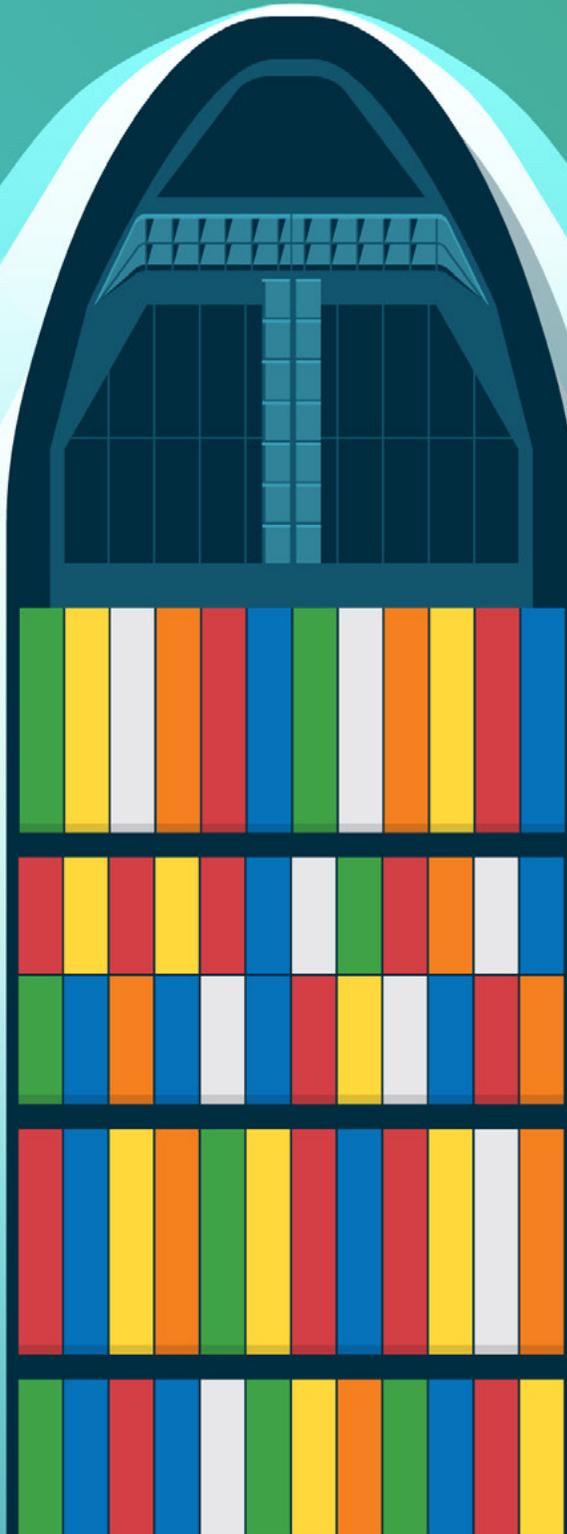


Introduction to CONTAINERS

eGuide



TECHWELL™



3

Why Choose Containerization?

5

Virtualization or Containerization? Choosing the Right Strategy

6

Deploying, Running Applications in Docker Containers

7

How Docker Enables Agile Software Development

12

Demystifying DevOps: A Day in the Life of a DevOps Tester

13

Exploring Containers: Creating a Dockerfile

14

How Testers Can Use Docker to Shift Left and Automate Deployments

15

Additional Resources

In the continuing search to accelerate development, test, and production cycles, organizations are looking more and more at the benefits of containerization. Containers help software run reliably when moved from one environment to another. Containers effectively bundle an entire runtime environment, rendering differences in OS distributions and underlying infrastructure non-issues. In addition, containers are lightweight, modular, portable, and require significantly less overhead than server or machine virtualization.

Introduced in 2013, Docker has become the industry standard to build and share containerized apps. This eGuide is filled with information and resources to help you explore the world of containers and whether or not they are right for your process.

In this Containers eGuide

Why Choose Containerization?

In recent years, containers have been adopted by many organizations. Why use containerization? What are the advantages that have spurred its adoption? Let's look at some of the factors that govern the choice of containerization.

Virtualization or Containerization? Choosing the Right Strategy

Virtualization and containerization are top two approaches when it comes to enabling scalability, limiting overhead costs, and standardizing software development, deployment, and management across multiple platforms. Careful selection of one strategy over another helps an IT team become more agile and responsive to ever-changing business needs. But how should they decide which better fits their requirements?

Deploying, Running Applications in Docker Containers

Deploying and running applications in Containers is one of the hottest trends in DevOps and IT today. Docker, a containerization platform that lets users easily package, deploy, and manage their applications within containers, is principally responsible for bringing containers to the mainstream. This article provides information about what containers are, their relationship with DevOps, and the benefits derived from container implementation.

How Docker Enables Agile Software Development

Docker has revolutionized how software is packaged, distributed, and deployed, so it's easy to see why it has become the de facto containerization platform. But have you thought about how Docker actually makes software development, testing, delivery, and deployment more agile? Let's look at how Docker inherently supports several of the founding principles of agile software development.

Demystifying DevOps: A Day in the Life of a DevOps Tester

The idea of working as a test specialist on a team using DevOps can feel intimidating. There are at least two technology stacks you need to be familiar with, add a source code repository like Git, a few test frameworks, and a scripting language, and you start to approach a useful skill set. This article outlines a normal day of testing in DevOps for the author.

Exploring Containers: Creating a Dockerfile

Docker is currently the most popular containerization platform. Let's look at what goes into creating a Dockerfile, which could be used to build a runnable Docker image.

How Testers Can Use Docker to Shift Left and Automate Deployments

There are multiple ways Docker can be to help with testing in continuous delivery and integration (CD and CI). This article outlines some of the ways to help facilitate that process.

Additional Resources

3

Why Choose Containerization?

5

Virtualization or Containerization? Choosing the Right Strategy

6

Deploying, Running Applications in Docker Containers

7

How Docker Enables Agile Software Development

12

Demystifying DevOps: A Day in the Life of a DevOps Tester

13

Exploring Containers: Creating a Dockerfile

14

How Testers Can Use Docker to Shift Left and Automate Deployments

15

Additional Resources

Why Choose Containerization?

By Deepak Vohra

In recent years, containers have been adopted by many organizations. While some other implementations of containers are available, Docker is probably the one you've heard of, as it has become the standard containerization platform.

Why use containerization? What are the advantages that have spurred its adoption? Let's look at some of the factors that govern the choice of containerization.

Lightweight Modules

Docker containers are lightweight units of software that run in isolation on a Docker Engine, which runs on an underlying operating system. Each container has its own networking and file system.

A single Docker container does not fully use an underlying operating system (OS), but makes use of a snapshot or a section of the operating system kernel. In this way, multiple Docker containers may run on the same OS.

A single virtual machine, in contrast, uses a complete operating system per application, and consequently, virtual machines may not be able to use the underlying OS fully, resulting in an underutilized OS system kernel. Infrastructure is better utilized with Docker containers.

Packaging and Dependency Encapsulation

A Docker image packages a complete software solution. A Docker container is an instance of a Docker image, which specifies the software to install, dependencies to install, and commands to run.

A single Docker container encapsulates all the dependencies needed to run an application, thus eliminating the need to install the dependencies separately. Software is easier to install and run using containers.

