



Containers 101

A container is a lightweight, portable approach to running multiple applications on the same operating system kernel. Applications are isolated and packaged only with their unique dependencies, allowing for increased density because containers consume fewer resources than traditional virtual machines.

Why containers?



Developers

Unlock ultimate productivity and freedom
Deploy as multi-tier distributed apps in IaaS or PaaS models, if needed



Operations

Provide standardized environments for development, QA, and production teams
Achieve higher utilization and compute density
Rapidly scale up or down to meet changing business needs



DevOps

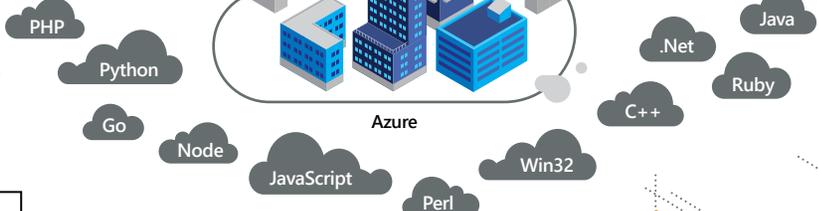
Integrate people, processes, and tools for optimized app development
Focus on standardized infrastructure
Allow developers to focus on building, deploying, and testing apps

Datacenter to cloud

On-Premises

Service Provider

Development framework and languages



Development tools



The key for developers is the ability to choose the right tool at the right time.



PowerShell



eclipse



Visual Studio

Leveraging these tools, containers enable:

- Rapid deployment
- Track changes / rollback
- Greater flexibility

Docker integration

Docker Hub:

Download a huge collection of open and curated applications.

Docker Engine:

Docker Engine for Windows
Server containers will be developed under the aegis of the Docker open source project.

Collaboration:

Bring Windows Server containers to the Docker ecosystem to expand the reach of both developer communities.

Docker client:

Windows customers can use the same Docker client and interface in multiple development environments.

Windows

Linux

docker

The technology

Container

- No virtualized hardware components
- Self-contained virtual instance with application and minimal OS components
- High resiliency due to abstraction
- Highly portable regardless of targeted host

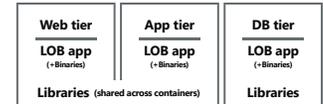


Virtual machine

- Fully virtualized set of abstracted hardware and drivers
- Full production OS with maintenance, patching, and security protocols
- Solutions installed as full applications
- Higher resource consumption
- Portability requires moving the entire virtual machine

Windows Server container

- Shares OS kernel
- Deployed and managed with Microsoft Visual Studio, Windows PowerShell, or Docker client



Windows User Mode
Windows Kernel
Physical/Virtual Server



Hyper-V container

- Isolated kernels with hypervisor to separate containers
- Deployed and managed with the same tools—Visual Studio, PowerShell, or Docker client

