

DevBox vs Codespaces

la battaglia per l'ambiente di sviluppo perfetto!





Massimo Bonanni Senior Technical Trainer @ Microsoft



Sponsor



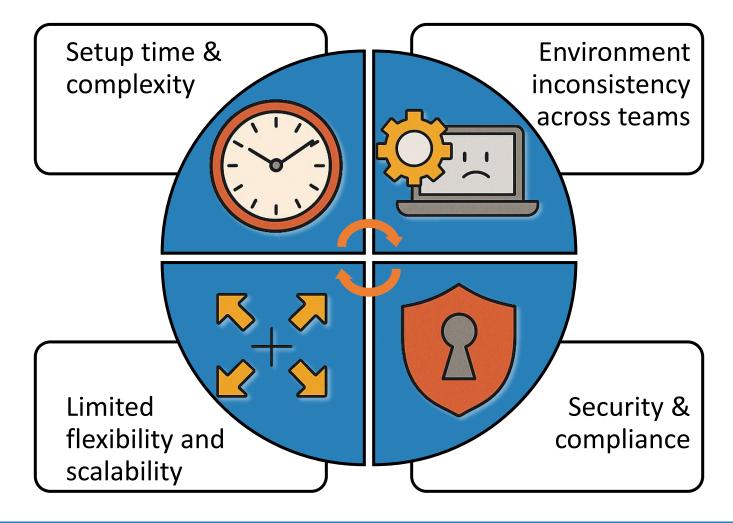








Challenges of Traditional Dev Environments















Challenges of Traditional Dev Environments



Days

- Acquire the physical workstation (e.g., laptop or desktop).
- Assign it to the developer (with asset tracking).

Purchase hardware

Initial Setup

- Unbox, power on, and run initial OS setup.
- Connect to corporate network.

Security Config



2-4 hours

- Install OS (e.g., Windows, Linux, macOS) and patches.
- Join the device to the corporate domain or identity provider (e.g., Entra ID).
- Set up user account and permissions.

OS Installations and

Development Tools Installation

- Install programming languages, compilers, SDKs (e.g., Java, .NET, Python).
- Install IDEs/editors (e.g., VS Code, IntelliJ, Eclipse).



1-2 hours

- Clone repositories and set up projectspecific configs.
- Set up environment variables, local servers, or databases as needed.

Configuration & Personalization



1-2 hours



2-4 hours















Cloud-Based Dev Environment Provisioning



1 hour

- Create or provision user account.
- Grant access to required cloud services, repositories, and environments.
- Enable SSO, MFA, or federated login.

Account & Access Setup

Environment Template Creation

- Define environment as code (e.g., devcontainer.json, Dockerfile).
- Specify base image, tools, dependencies, and scripts.

1-3 hours (one-time effort)



2-10 mins

- Assign appropriate CPU, memory, storage (often configurable).
- Spin up a new cloud environment (e.g., Codespace, container, VM).
- Automatically pull code, build environment using templates.

Provisioning & Launch

Tooling & Extensions Setup

- IDE (often VS Code Web), languages, frameworks already installed from template.
- Auto-install extensions and plugins from config.



15-30 mins

15-30 mins

- Load personal dotfiles, aliases, editor themes.
- Configure environment variables, shell preferences.
- Sync tools or settings between local and cloud (if hybrid setup).

Configuration & Personalization















Microsoft DevBox





Introducing Microsoft DevBox



Cloud-based solution providing ready-to-code workstations



Offers self-service access to pre-configured "dev boxes"



Optimized for the specific needs of development teams



Reduces the time and complexity of environment management



Allows developers to focus on coding and building applications















Prerequisites



An Azure account with an active subscription.



Owner or Contributor role.



Microsoft Entra ID.



User licenses for Windows 11/10 Enterprise and Microsoft Entra ID.



Register the *Microsoft.DevCenter* resource provider





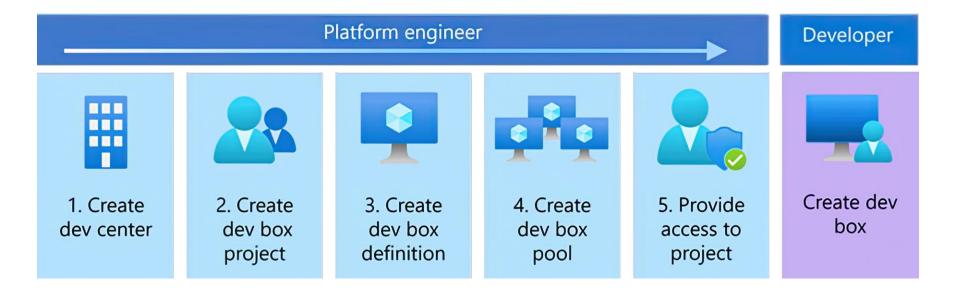


























dev center

A **dev center** is the top-level resource for Microsoft Dev Box.

A dev center contains the collection of projects and the shared resources for these projects, such as dev box definitions and network connections.

There's no limit on the number of dev centers that you can create, but most organizations need only one.

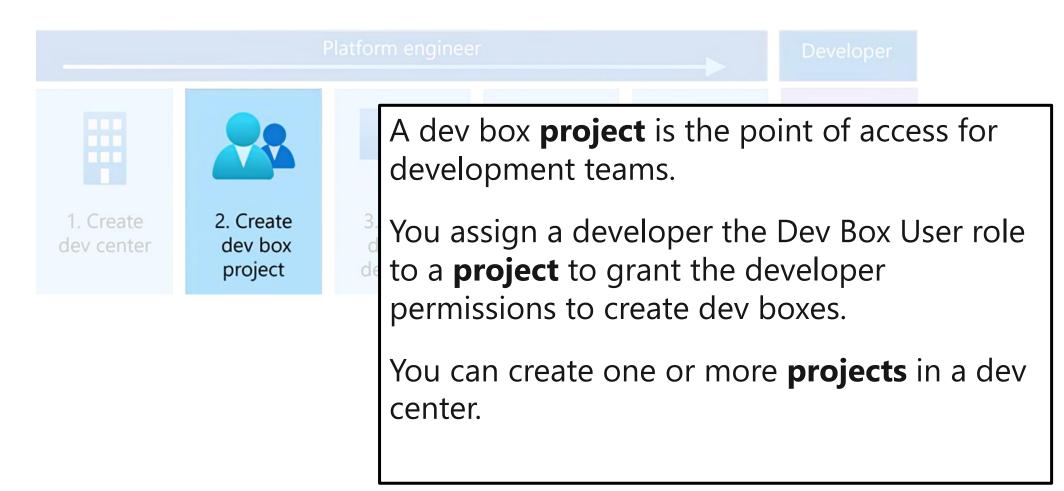














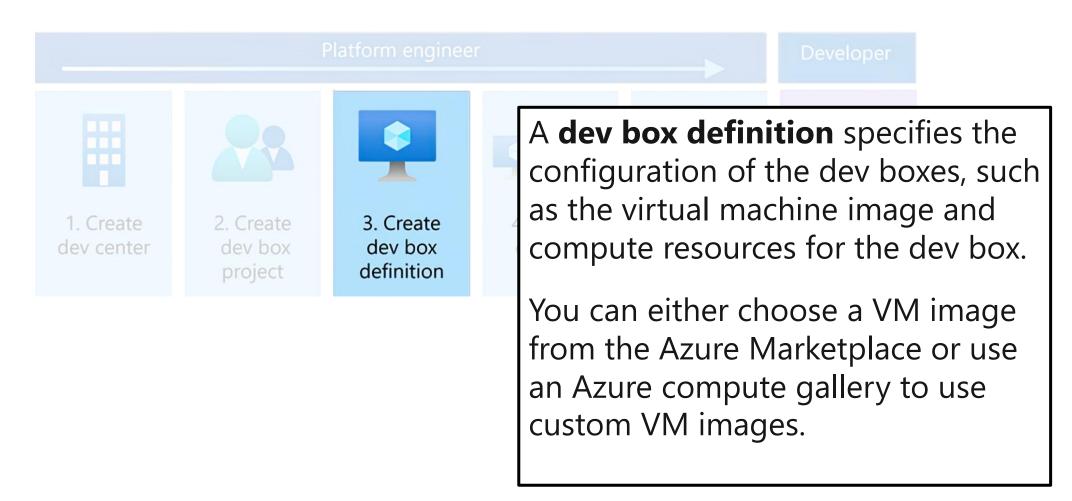
























A dev box pool specifies the configuration for dev boxes, such as the dev box definition, the network connection, and other settings.

All dev boxes that are created from a dev box pool share the same configuration.













Use **RBAC** to provide access to all the resources of DevBox based on their organizational role types:

- Platform engineer: influence permissions for dev centers, catalogs, and projects
- Dev manager: influence permissions for project-based resources
- Developer roles: influence permissions for users



5. Provide access to project



Create dev













They might choose from a specific pool based on the VM image, compute resources, or the location where the dev box is hosted.

Once the **dev box** is running, dev box users can remotely connect to it from the developer portal.

Dev box users have full control over the dev boxes they create and can manage them from the developer portal.



Create dev box





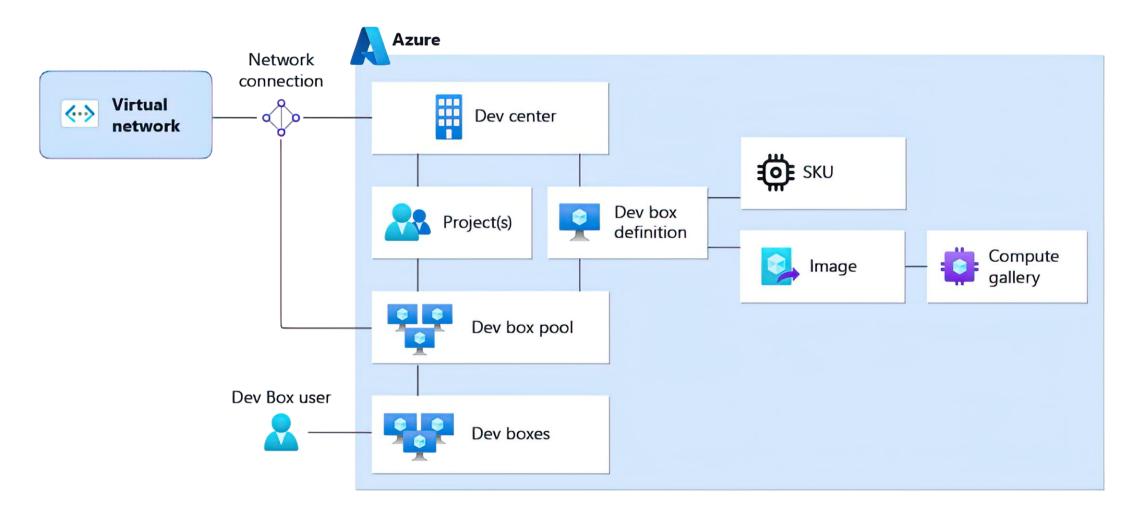








Microsoft DevBox Architecture

















Demo Microsoft DevBox



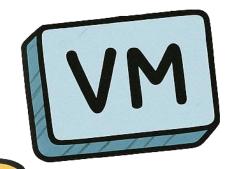


Pricing











Microsoft Dev Box uses a pay-as-yougo pricing model.

Costs are split into two components:

- storage costs, which are incurred as long as the Dev Box exists
- compute costs, which apply only when the Dev Box is in use (running).

You can reduce costs by automatically stopping Dev Boxes when not in use.

	Pricing per Dev Box Instance		~
sкu	Max Monthly Price	Hourly Compute	Monthly Storage
8 vCPU, 32 GB RAM, 256 GB Storage	€138.23	€1.50	€19.01
8 vCPU, 32 GB RAM, 512 GB Storage	€157.23	€1.50	€38.01
8 vCPU, 32 GB RAM, 1,024 GB Storage	€195.24	€1.50	€76.02
8 vCPU, 32 GB RAM, 2,048 GB Storage	€271.26	€1.50	€152.03
16 vCPU, 64 GB RAM, 256 GB Storage	€257.45	€2.99	€19.01
16 vCPU, 64 GB RAM, 512 GB Storage	€276.46	€2.99	€38.01
16 vCPU, 64 GB RAM, 1,024 GB Storage	€314.46	€2.99	€76.02
16 vCPU, 64 GB RAM, 2,048 GB Storage	€390.48	€2.99	€152.03
And have	15	15	
B	15	15 B	55











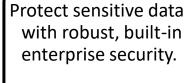




Benefits

Deploy ready-to-use, cloud-based developer environments in minutes.

Rapid Provisioning



Enterprise-Grade Security



Maintain uniform configurations that eliminate setup discrepancies.

Consistency Across **Environments** Scale and customize resources effortlessly to match project demands.

Scalability & Flexibility



Optimize spending by switching from heavy hardware to a pay-per-use model.

Cost Efficiency



Oversee all developer environments seamlessly from a single control console.

Centralized Management



Enjoy effortless integration within the broader Microsoft ecosystem.

Seamless Integration



















GitHub Codespaces





What are GitHub Codespaces?



Cloud-based IDE: Instantly launch fully managed coding environments directly from GitHub.



Customizable and Consistent: Uses containers (.devcontainer) to standardize developer setups.



Integrated with Azure: Built on Azure infrastructure, simplifying integration with Azure services.



Improves Collaboration: Easy sharing and collaborative coding without local setup hassles.



Secure and Scalable: Provides isolated, secure environments with automatic scalability.









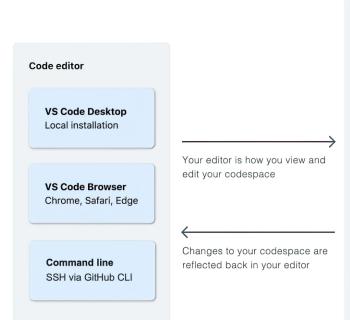


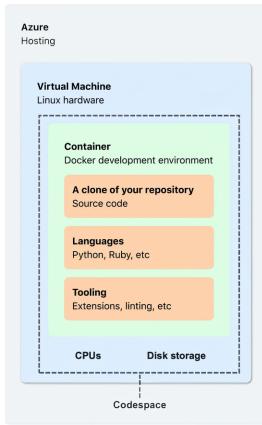




How they work

- Each Codespace you create is hosted by GitHub in a **Docker container**, running on a **virtual** machine.
- You can choose from a selection of virtual machine types.
- By default, the Codespace development environment is created from an Ubuntu Linux but you can use an image based on a Linux distribution of your choice.
- Windows and macOS are not supported operating systems

















Where you can use them



From a GitHub template or any template repository on GitHub to start a new project



From a branch in your repository for new feature work



From an open pull request to explore work-in-progress



From a commit in a repository's history to investigate a bug at a specific point in time













The Codespace creation process

A dedicates VM launches.

Full access to computing resources and root access.

Repository cloned into VM's /workspaces directory and mounted in the dev container.

> VM and storage are assigned to your Codespace

Dev container is created

Codespaces uses Docker containers configured via devcontainer.json.

GitHub uses a default image if no configuration is provided.

Default image details are in the devcontainers/images repository

You can connect to it by using:

- Your web browser
- Visual Studio Code
- GitHub CLI

Connecting to the Codespace

Post-creation setup

After connecting, your codespace runs automated setup commands defined in devcontainer.json, like postCreateCommand and postAttachCommand.

















Demo GitHub Copilot





Pricing

- **GitHub Codespaces** utilizes a **usage-based** pricing system, charging for **compute time** starting at \$0.18 per hour for a 2-core machine and \$0.07 per GB of **storage** monthly.
- Personal accounts have monthly usage limits—
 15 GB storage and 120 core hours for Free, and
 20 GB storage and 180 core hours for Pro.
- Organizations need a non-zero spending limit to use Codespaces and can manage costs by selecting machine types and usage policies.

Machine type	Unit of measure	Included usage multiplier	Price
2 core	1 hour	2	\$0.18
4 core	1 hour	4	\$0.36
8 core	1 hour	8	\$0.72
16 core	1 hour	16	\$1.44
32 core	1 hour	32	\$2.88
Storage	1 GB-month	Not applicable	\$0.07















Benefits

Launch fully configured dev environments from your GitHub repo in seconds—no local setup needed.

Instant Environments Use devcontainer.json for your environment, eliminating "it works on my machine" issue.

Consistent & Reproducible Setups



New team members can start immediately, without installing tools or configuring dependencies.

Simplified Onboarding



Offload heavy compute tasks to the cloud ideal for lower-end devices or resourceintensive projects.

Cloud-Based Resource Efficiency



Work directly with pull requests, branches, and issues in a unified, GitHub-native workflow.

Seamless GitHub Integration



Access your environment securely via browser or VS Code desktop from any device.

Flexible Access from Anywhere



Each Codespace runs in its own container, ensuring isolation, security, and minimizing conflicts.

Secure & Isolated **Environments**



















Who win?





DevBox vs Codespaces...who win?



Choose Microsoft Dev Box if



Choose GitHub Codespaces if

You need a **persistent** and **secure** development environment.

You prefer a lightweight, ephemeral environment for rapid development.

Your organization requires compliance with industry standards. You're working on open-source projects or need to collaborate quickly.

You're developing applications that rely on Windows-specific tools or configurations.

You want to **onboard** new developers with minimal setup time.





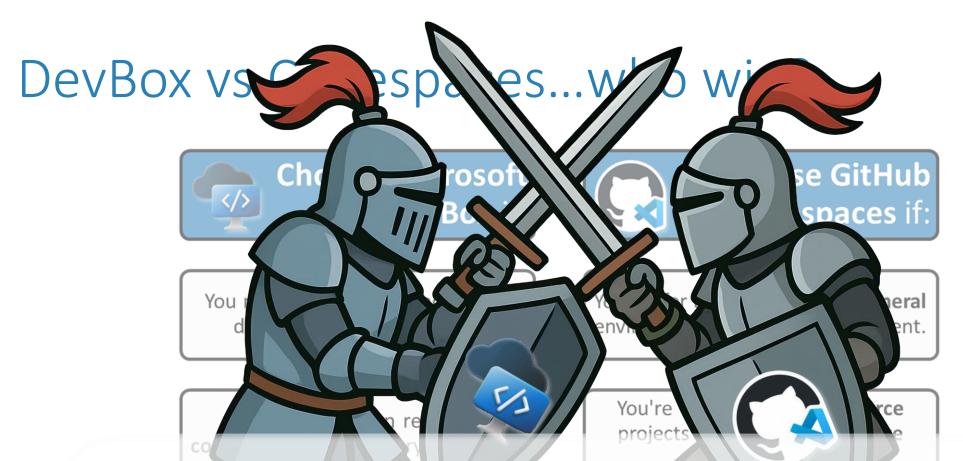












Not every battle ends with a winner! Sometimes, it's about choosing the right weapon for the right war.



Any Questions?



Massimo Bonanni

Senior Technical Trainer massimo.bonanni@microsoft.com @massimobonanni













