Brought to you by

Coder

Cloud Development Environments

for dummies®

A Wiley Brand

Coder Special Edition

Recognize the benefits of CDEs

Choose the right CDE for your organization

Implement and integrate your CDE

Matt Vollmer

About Coder

Coder delivers self-hosted cloud development environments consistently provisioned as code and preconfigured for developer activity on day one. Preferred by enterprises, Coder is open-source and runs on-premises or in your cloud, giving software and model developers access to powerful infrastructure without compromising governance.

Simply put, Coder moves development environments and source code from developers' laptops to your centralized infrastructure, where they can access remote environments via their favorite desktop or web-based IDE. This is the optimal intersection of improved developer experience, productivity, and security.

Cloud Development Environments

Coder Special Edition

by Matt Vollmer
Coder Head of Product Marketing

for dummies®
A Wiley Brand

Cloud Development Environments For Dummies®, Coder Special Edition

Published by
John Wiley & Sons, Inc.
111 River St.
Hoboken, NJ 07030-5774
www.wiley.com

ISBN 978-1-394-35801-4 (pbk); ISBN 978-1-394-35802-1 (ebk);
ISBN 978-1-394-35803-8 (ebk)

Publisher's Acknowledgments

Editor: Elizabeth Kuball
Acquisitions Editor: Traci Martin
Senior Managing Editor: Rev Mengle
Client Account Manager: Jeremith Coward
Production Editor: Tamilmani Varadharaj
Special Help: Joe Kraynak

Table of Contents

Introduction

Cloud development environments (CDEs) are changing how software gets built. Instead of juggling complicated local tools or dealing with overprovisioned virtual machines, CDEs provide a centralized space where developers can work from anywhere. They bring together all the tools, code, and infrastructure you need into one convenient place. It's like having your entire workspace in your back pocket, literally — you can even code from your iPad in a pinch.

This book helps you make sense of it all. It explains what CDEs are, how they work, and why they're gaining traction with teams of all sizes. You'll discover how to evaluate different types of CDEs, so you can find the one that fits your team's needs and start taking the first steps toward rolling out your own CDE successfully.

You're already interested in CDEs, so I'm not here to sell you on the idea. Instead, I break things down in a way that's easy to follow, giving you the information you need without overloading you with superfluous details or cryptic jargon.

Foolish Assumptions

Here are a few assumptions I've made about you as a reader:

- You're a platform engineer, developer, or data scientist — or someone who works in the same field.
- You've used or managed development environments before, whether those were on local machines, virtual machines, or virtual desktop infrastructure.
- You're open to exploring new ways to work. CDEs enable developers to work anywhere on any machine without forcing them to change the tools they already know and love.

Icons Used in This Book

Throughout this book, you'll see a couple of handy icons to help you along the way. Here's what they look like and what they signify:

REMEMBER

This book is short enough to zip through, but make sure to pay attention to paragraphs marked with the Remember icon. They highlight the key ideas worth keeping in mind.

TIP

The Tip icon guides you to CDE best practices along with faster, easier ways to perform a task.

Where to Go from Here

This book is only a few dozen pages long, so it just scratches the surface of cloud development environments and how they can improve security and developer productivity in your organization.

If you finish this book with an appetite for more information, here are a few resources you may want to check out:

- **An Enterprise Buyer's Guide to Cloud Development Environments:** `https://coder.com/ebooks-and-reports/ebooks/an-enterprise-buyer-s-guide-to-cloud-development-environments`
- **Cloud Development Environment Maturity Model:** `https://coder.com/ebooks-and-reports/ebooks/cloud-development-environment-maturity-model`
- **Cloud Development Environment Blog: CDE 101 (a blog series on CDE fundamentals):** `https://coder.com/blog?category=cde-101`

IN THIS CHAPTER

- Understanding what a CDE is and exploring its key features
- Recognizing the shortcomings of traditional development environments
- Considering the advantages of CDEs
- Recognizing the difference between a CDE and an IDE
- Busting common myths and misinformation

Chapter 1
Grasping Cloud Development Environment Fundamentals

If you've ever struggled with outdated or inconsistent development environments, you're aware of their shortcomings. Traditional development environments can be slow, complicated, and inflexible. Cloud development environments (CDEs) offer a solution.

This chapter lays the groundwork for understanding CDEs. It covers what CDEs are and how they compare to traditional approaches, and it addresses a few misconceptions along the way. By the end of this chapter, you'll have a clear understanding of what makes CDEs so valuable and how they fit into modern development workflows.

Explaining CDEs

A *CDE* is a fully managed environment hosted on your cloud or on-premises infrastructure where developers and data scientists can code applications or train machine learning (ML) models. It's where all the tools, libraries, and infrastructure you need come together, making it easy to provision new development environments and work from anywhere. Think of it as your development workspace, but one that doesn't tie you to a specific location or machine.

Recognizing the Challenges of Traditional Solutions

Development environments have come a long way, but many traditional solutions still have significant downsides:

- **Local development environments:** This old-school approach has every developer managing their environment on their own personal machine. Although it gives you full control, it's also a hassle. Configuring tools and dependencies is time-consuming, troubleshooting inconsistencies can slow projects down, and scaling this process across a team gets messy fast.
- **Virtual desktop infrastructure (VDI):** Some companies have moved to VDIs to centralize environments, but VDIs come with their own challenges. They're expensive, they're often slow, and they don't provide the best developer experience. Laggy performance can frustrate developers, and maintaining these systems is no small task.
- **"Shadow" virtual machines (VMs):** Developers sometimes use ad hoc VMs to bypass the headaches of local environments or VDIs, but these "shadow" solutions often lack governance. They're frequently overprovisioned, leading to wasted resources, and their lack of oversight can introduce security risks.

Appreciating the Benefits of CDEs

CDEs aren't a brand-new concept — they've been around for a while. However, recent advances in technology, particularly in networking performance, have made them viable for teams of all sizes. Here's why they're worth considering:

- **Faster onboarding:** With preconfigured environments, new team members can get up and running in minutes rather than days. CDEs automate the tedious steps that developers previously had to manually follow using README files.
- **Secure code access:** Code and data stay centralized on your infrastructure — not developers' laptops — reducing the risks of loss or theft associated with local environments.
- **Support for remote work:** Whether your team is fully remote or spread across offices, CDEs allow developers to access their environments from anywhere with thin client laptops.

By addressing the weaknesses of traditional approaches, CDEs offer a modern alternative that improves both productivity and security. See Chapter 2 for more about the benefits of CDEs for teams of any size and Chapter 6 for a list of ten ways CDE's improve productivity and the developer experience.

Examining the Key Features of CDEs

Although every development platform is unique, CDEs have a few features that make them special, including the following:

- **Automatic provisioning:** Environments consistently provide developers with the right tools, libraries, and infrastructure for specific projects or applications.
- **Centralized management:** Authorized team members can control everything from a single interface, making global workspace updates and patches much easier.
- **Scalable resources:** Developers can scale up compute and storage resources for tasks such as testing, debugging, and training ML models.
- **Access control:** Only authorized individuals can access sensitive code and data.

Distinguishing a CDE from an IDE

REMEMBER

CDEs and integrated development environments (IDEs) share many traits, but they're not the same thing.

An *IDE*, such as IntelliJ, Visual Studio, or VS Code, is a tool for writing, debugging, and testing code. In contrast, a CDE is a platform that provisions and manages the entire development environment. It provides the infrastructure, libraries, and configurations needed to support your work.

The two complement each other. In fact, you can connect to your CDE from your existing IDE to create a consistent, centralized workspace that gives you access to all the tools you already know and love. Extensions for IDEs such as VS Code and IntelliJ make it easy to connect to a CDE, delivering a seamless experience that feels just like working locally.

Clearing Up Common Misconceptions about CDEs

CDEs are gaining traction, but some misunderstandings are still out there slowing the pace of CDE adoption. Here are a few of the more common misconceptions and the facts that prove them wrong:

- **"CDEs replace your favorite tools."** Not true! CDEs work with the tools you already love, whether that's your favorite IDE or a version control system.
- **"I can just use Devcontainers instead."** Devcontainers are helpful, but they don't provide the same centralized management, scalability, or security features that a CDE offers. Devcontainers are just a spec, but you still need a place to run them.
- **"They're slow and too complex."** Thanks to advances in networking and cloud technology, modern CDEs are fast and user-friendly, giving you a local-like experience without the headaches.

REMEMBER

CDEs offer a way to work smarter, not harder, by combining the flexibility of the cloud with the tools you already use. They enable you to balance flexibility, security, and developer productivity without sacrificing one to achieve the other.

IN THIS CHAPTER

- Discovering the ways CDEs can improve developer productivity
- Looking into how CDEs enhance security
- Understanding how CDEs can reduce development costs

Chapter 2
Recognizing the Benefits of CDEs for Teams of Every Size

As development teams grow, so do their challenges. From scaling developer productivity to improving security and controlling costs, every team — whether a small startup or a global enterprise — faces these common hurdles. That's why cloud development environments (CDEs) are gaining traction. Although their benefits are often associated with large-scale teams, even smaller teams can gain significant advantages.

That said, CDEs aren't typically useful for indie developers or hobbyists. You feel the full impact of CDEs when you're managing multiple developers or dealing with complex infrastructure and project requirements. Teams exploring CDEs often cite developer experience, governance, and security as the key factors driving their decision. This chapter shows you how CDEs deliver in all those areas, with the added benefit of reducing costs.

Boosting Developer Productivity

REMEMBER

For development teams, time is everything. CDEs fundamentally transform how teams operate by removing common bottlenecks such as complicated project onboarding processes, mismatched environments among developers, and resource constraints. They simplify how environments are provisioned, updated, and maintained, giving developers more time to focus on building instead of troubleshooting. Specifically, CDEs deliver the following productivity benefits:

- **Faster onboarding for developers and smoother project transitions:** Onboarding new developers or transitioning existing ones to new projects can be painfully slow with traditional local environments. Developers often switch projects multiple times a year, and provisioning new environments can take weeks due to dependency complexities and approval processes. With CDEs, all that goes away. Developers simply log into their preconfigured CDE via a link, and they can start working immediately.
- **Simplified environment provisioning and updates:** CDEs make managing environments far easier. Platform engineers can roll out updates globally to workspaces shared by multiple developers on the same project. This process ensures that every developer is working in an environment that's consistent and up-to-date, and it all happens at the same time, reducing manual intervention and configuration drift.
- **Access to powerful cloud resources:** By leveraging high-performance cloud resources for tasks such as builds and machine learning (ML) model training, developers free up their local machines for other work. No more bricking their laptops while waiting for resource-intensive processes to complete.

Improving Security

Security is at the core of why many teams adopt CDEs. By centralizing code, data, and tools in the cloud, CDEs significantly reduce risks tied to decentralized development

environments on developers' laptops and unapproved virtual machines (VMs). They provide better control over who has access to what, ensuring that intellectual property (IP) is protected and compliance requirements are met. For teams working with sensitive data or in regulated industries, CDEs offer peace of mind that's hard to match. Features that improve security in the CDE include the following:

- **Centralized source code and data:** CDEs can be hosted in a cloud environment — whether that's Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure — or even on on-premises infrastructure, including air-gapped environments that are completely detached from the internet. This flexibility makes CDEs especially valuable for government agencies and highly regulated enterprises.
- **Protecting IP:** CDEs help reduce exfiltration risks from developer laptops. Developers working with real-world customer or patient data, for instance, often need to pull that data onto their laptops for tasks such as training ML models. With CDEs, both the source code and the data remain colocated in secure cloud or on-premises infrastructure alongside the compute resources needed for building or training those models, ensuring that sensitive data never leaves the secure environment.
- **Auditing and control over tools:** CDEs provide deeper auditing capabilities, enabling platform and security teams to see exactly how developers are using their workspaces and what actions they're taking. Unlike local development, where activity is much harder to monitor, CDEs give teams the information they need to investigate and react to potential issues in a timely manner.

Saving Time and Money

Cost-efficiency isn't typically the first reason a team evaluates a CDE — developer experience and security are usually the main drivers. However, cost-efficiency is an incredibly valuable by-product. By reducing reliance on expensive local hardware, providing better control over cloud resources, and rationalizing tool usage, CDEs help teams optimize their budgets without

compromising performance. (See Chapter 4 for more about rationalizing tool usage.)

For organizations looking to scale more cost-effectively, CDEs can help in the following ways:

- **Cutting back on high-end local hardware:** By running resource-intensive processes on cloud infrastructure, CDEs allow developers to use more cost-effective thin client machines instead of high-powered laptops or desktops. These devices are far less expensive, last longer, and don't need to be replaced as frequently.
- **Optimizing cloud usage with automation:** CDE features such as autostop/autostart ensure that unused environments aren't consuming unnecessary resources, while bin-packing multiple developer workspaces onto a single VM helps reduce overall infrastructure costs without sacrificing performance. Additionally, resource quotas enable platform teams to control how much infrastructure developers can provision, protecting organizations from runaway cloud costs.
- **Efficiency of human capital:** CDEs also save time for developers. By onboarding and transitioning between projects more efficiently, teams save countless hours and weeks every year that otherwise would be wasted.

CDEs bring a powerful combination of productivity, security, and cost-efficiency to midsize and enterprise-scale teams. They simplify how developers work, enhance security by centralizing code and data on protected infrastructure, and offer tangible savings by optimizing hardware and cloud usage.

REMEMBER

CDEs also present a modern, viable alternative to legacy virtual desktop infrastructure (VDI). Not only are VDIs expensive to license, but they require significant overhead to manage. For enterprises already running VDI for developers, CDEs can deliver material cost savings while providing a far better developer experience.

IN THIS CHAPTER

- Knowing what to look for in a CDE
- Checking out the leading solution providers
- Avoiding common mistakes

Chapter 3
Choosing the Right CDE

Selecting the right cloud development environment (CDE) for your team can be a game changer — or a costly misstep if done without sufficient research and planning. Whether you're a small startup, a large enterprise, or a highly governed agency, choosing a CDE that aligns with your team's unique needs and goals is critical.

This chapter guides you through the key evaluation criteria, maps out the CDE landscape, and highlights common pitfalls to avoid so you can make an informed and discerning decision.

Establishing Key Evaluation Criteria

REMEMBER

When evaluating CDEs, considering how the platform's features align with your organization's requirements is essential. Here are some of the most important criteria to focus on:

- **Open-source versus proprietary platform:** Does your organization prefer the flexibility and transparency of open-source or the packaged convenience of a proprietary solution?
- **Hosting and maintenance model:** Are you looking for a self-hosted, self-managed solution that your team maintains and has total control over or a vendor-hosted and -managed

solution — a platform as a service (PaaS) that removes some of the operational burden?

- **Data sovereignty:** Does the platform let you control the regions where your CDEs and their data are hosted? This consideration is critical for organizations with strict compliance requirements.
- **One hundred percent offline/air-gapped support:** Does the platform offer air-gapped capabilities for organizations needing to operate in isolated environments, such as government or defense agencies?
- **Cost model:** Does the pricing structure align with your team's budget and scalability plans?
- **Multicloud support:** Can the platform run across different cloud providers, or does it lock you into one ecosystem?
- **Workspace platform:** Does the platform support the types of environments your developers need — Docker, Kubernetes, or virtual machines (VMs)?
- **Workspace extensibility:** Can you customize developer environments with the tools and configurations your team requires?
- **Workspace operating system support:** Does the platform support the operating systems your developers rely on, such as Linux, macOS, and Windows?
- **Workspace tooling:** How well does the platform integrate with integrated development environments (IDEs), version control systems, authentication platforms, and monitoring tools?


TIP

Map out your team's existing IDEs, tools, languages, and workflows to ensure compatibility. Choosing a platform that aligns with your existing ecosystem is essential for a smooth implementation.

Mapping the CDE Landscape


REMEMBER

Every organization has unique requirements for a CDE, and the growing market offers a variety of options tailored to different needs. Smaller teams without stringent security or governance demands may favor software as a service (SaaS) platforms

hosted on managed infrastructure, because these often reduce operational complexity and provide easy access to preconfigured environments. In contrast, larger enterprises and highly regulated industries typically require platforms with stronger security, greater extensibility, and seamless integration into their existing systems.

Organizations evaluating CDEs often consider solutions from the following providers:

- **Coder** provides a self-hosted, self-managed CDE platform that appeals to enterprises needing hosting flexibility, whether in public cloud, hybrid, or fully air-gapped on-premises environments. It supports IDEs and operating systems without locking teams into a specific ecosystem. Its adoption among enterprises is often driven by its extensibility, multicloud support, and the ability to integrate seamlessly with existing tools and workflows.
- **Amazon CodeCatalyst, GitHub Codespaces, and Microsoft Dev Box** cater primarily to small to medium-size teams, particularly those already embedded within the Microsoft or Amazon Web Services (AWS) ecosystems. However, their tight integration with their respective ecosystems may present challenges for organizations requiring multicloud flexibility or additional customization.
- **Gitpod** is another option that appeals to small to midsize teams. Its vendor-managed model simplifies operations but limits control for teams needing more flexibility, particularly enterprises that require fine-grained control over their environments or the ability to operate across different clouds beyond AWS.

The CDE landscape spans a broad range of use cases, making it essential for organizations to align their choice with their size, needs, and priorities:

- **Startups and small teams:** Lightweight, managed solutions can be an excellent fit for small teams focused on rapid deployment and minimal maintenance overhead.
- **Large enterprises:** Scalability, governance, and platform control are often nonnegotiable. Self-hosted or self-managed platforms tend to align better with these requirements.

- **Highly regulated industries:** Air-gapped environments, compliance-focused features, and deep customization capabilities are critical for teams in government, health care, or finance.

Steering Clear of Common Pitfalls When Choosing a CDE

REMEMBER

Selecting a CDE requires careful consideration to avoid poor choices that can hinder adoption and long-term success. Here are the key pitfalls to watch out for and guidance on how to avoid them:

- **Overlooking scalability:** You may plan to start small, but think about how many developers could be using the platform in the next 6 to 12 months. Choosing a solution that can't scale with your team's needs could lead to costly and disruptive changes later.
- **Ignoring developer feedback:** Surveying developers during the proof-of-concept process is essential to uncovering nuances you may not have considered. Understanding their workflows and specific pain points ensures that the CDE you choose improves their productivity instead of introducing unnecessary friction.
- **Underestimating integration challenges:** Fully understanding the tools, languages, and dependencies your developers rely on is essential before committing to a CDE. A solution that doesn't support these needs or that forces developers to overhaul their workflows can slow adoption and cause frustration.
- **Neglecting performance considerations:** Modern platforms can now deliver network performance that feels nearly identical to local development, with imperceptible latency. Make sure the CDEs you evaluate can support globally distributed teams with features such as proxies to ensure a consistent, high-performance experience.

IN THIS CHAPTER

- Highlighting the benefits to developer productivity and morale
- Showcasing efficiencies in the development environment
- Proposing a CDE as a way to enhance security
- Estimating the potential cost savings of a CDE

Chapter 4
Making the Business Case for a CDE

Convincing leadership to invest in a cloud development environment (CDE) requires a well-articulated business case that demonstrates its tangible benefits. This chapter explains how to present the business case for a CDE by focusing on its positive impact on developer productivity, cost savings, and enhanced security.

Keeping Developers Happy and Productive

REMEMBER

At the core of the business case for a CDE is the dramatic boost it can deliver to developer productivity and morale. Benefits include the following:

- **Streamlined onboarding:** Traditional local environments often require days to weeks of configuration before new hires can contribute code. With a CDE, developers can access a preconfigured environment in minutes, enabling them to hit the ground running.

- **Quick and easy project switching:** Developers often move between projects multiple times a year, and each transition typically involves setting up new environments, resolving dependency issues, and awaiting approvals. With a CDE, developers can switch projects simply by selecting a link to their new workspace. The time saved not only accelerates delivery but also reduces frustration, enhancing overall job satisfaction.


TIP

  Calculate, on average, how long it takes developers to switch projects throughout the year. This metric alone is often enough to justify investing in a CDE.
- **Reduced build times:** By leveraging powerful cloud resources, builds that once took hours can now be completed significantly faster. This efficiency gives developers more time to focus on coding and problem-solving, instead of waiting for resource-intensive tasks to complete.
- **Improved developer retention:** High-performing teams supported by effective tools are less likely to experience burnout and turnover. Considering that the cost of replacing a developer is roughly equivalent to a full-time salary, reducing churn can lead to significant long-term savings in recruiting and onboarding.

Rationalizing Tools and Optimizing Resources

CDEs bring cost-efficiency to several areas of the development process, including the following:

- **Rationalizing tools:** Hosting development environments in the cloud eliminates the need for expensive local tools and licenses for managing and maintaining local setups. This consolidation reduces tooling overhead while maintaining developer functionality.
- **Optimizing compute resources:** Traditional development environments often involve over-provisioning, such as when each developer is allocated their own virtual machine (VM) that runs 24/7 — even when it's not in use. CDEs solve this inefficiency by hosting multiple developer workspaces on a

single VM or Kubernetes pod, reducing the overall infrastructure footprint. Additionally, autostop ensures that resources aren't running during periods of inactivity, further lowering costs.

- **Reducing security overhead:** Securing local development environments requires significant effort from IT and security teams, including patch management, vulnerability scans, and device-specific configurations. With CDEs, these environments are hosted in the cloud, eliminating much of the manual overhead while maintaining strong security controls.

Mitigating Risks

CDEs mitigate risks significantly by centralizing code, data, and compute resources in a secure environment. This approach reduces vulnerabilities associated with decentralized development on individual laptops, such as unauthorized access, accidental data loss, and exfiltration. Enterprises that handle sensitive data, such as customer or patient information, can securely store and process that data alongside the development environment, ensuring that it never leaves the organization's infrastructure.

REMEMBER

Quick response to vulnerabilities is another key benefit. With CDEs, security teams can push updates or changes to all developer environments simultaneously, eliminating the delays and inconsistencies that often occur with manual updates on local machines. Timely updates ensure that critical security patches are applied instantly across the organization, reducing exposure to potential breaches.

The financial impact of these improvements can be significant. According to IBM's 2023 study, the average cost of a data breach is $4.5 million. Even if a CDE reduces the likelihood of a breach by a small percentage, the potential savings are substantial. Beyond cost avoidance, reducing risks tied to insider threats, stolen credentials, or unpatched vulnerabilities can protect an organization's reputation and maintain trust with customers and stakeholders.

By enforcing strict access controls, increasing visibility into developer activity, and reducing the attack surface, CDEs help

organizations strengthen their overall security posture and mitigate risks that could otherwise lead to severe financial and operational consequences.

Quantifying ROI for Leadership

REMEMBER

To build a compelling business case, translate the benefits of CDEs into tangible metrics that resonate with leadership, such as the following:

- **Developer onboarding time:** If onboarding is reduced from two weeks to one day, calculate the value of those saved hours across the team.
- **Reduced churn:** Estimate the cost savings of retaining developers by estimating the number of them that likely would have left due to frustration with current tools and inefficiencies. For every retained developer, you save roughly one full-time salary.
- **Infrastructure optimization:** Highlight the reduced costs of buying and maintaining less expensive on-premises hardware, running fewer VMs, and minimizing idle resources.
- **Tool rationalization:** Identify licenses and tools that can be eliminated or replaced and calculate the total annual cost savings.
- **Incident response improvements:** Demonstrate how faster updates and patching reduce the likelihood of breaches and compliance fines, which could otherwise cost millions.

REMEMBER

A good CDE provider should work with you to craft a tailored value hypothesis or return on investment (ROI) analysis that aligns with your organization's specific metrics and goals. By leveraging your unique data, they can help you clearly demonstrate the tangible benefits of adopting a CDE to your leadership team.

IN THIS CHAPTER

- Putting the right people in place
- Adjusting your approach to larger teams
- Preparing developers for the transition
- Smoothing the transition to the new CDE
- Encouraging and managing wider adoption

Chapter 5
Implementing a CDE

The process of implementing a cloud development environment (CDE) may seem daunting, but starting small and collaborating closely with developers can pave the way for a smooth rollout. This chapter outlines the key steps to implementing a CDE, from initial pilots to scaling up for enterprise use, and offers tips on overcoming common roadblocks and maximizing developer satisfaction along the way.

TIP

Start by running a small pilot project with a trusted team to gather feedback and refine your approach before rolling it out broadly. Don't try to force adoption. Developers are more likely to embrace a platform when they opt in, as opposed to having the change imposed on them.

Building Your CDE Implementation Team

The first step toward successful implementation involves building a small team of qualified individuals who support transitioning to a CDE. For midsize to enterprise companies, several key roles are critical for standing up a CDE deployment:

- **Platform engineer:** This role is essential for managing infrastructure as code (IaC) and implementing GitOps workflows. A platform engineer's expertise in scripting and

automation ensures that environments are consistently provisioned and updated.

- **DevOps or site reliability engineer (SRE):** With skills in Docker, Kubernetes, and Terraform images, the DevOps or SRE professional configures the technical foundation of the CDE. They're responsible for integrating the platform with existing infrastructure and ensuring scalability and reliability.
- **Product manager:** This person plays a critical role in identifying use cases and requirements, gathering developer feedback, and coordinating end-user enablement. The product manager ensures that the deployment aligns with organizational goals and addresses the needs of the developer community.

In addition to these core roles, the following supporting roles are key to a successful deployment:

- **Security personnel** help to ensure that the CDE adheres to company security policies and compliance requirements, including proper authentication, data handling, and monitoring.
- **Development team leadership** provides guidance on developer needs and advocates for adoption across teams.
- **Infrastructure personnel** ensure that the CDE integrates with the organization's cloud or on-premises infrastructure and can operate efficiently at scale.
- **Database personnel** assist with configuring database connections and ensuring that the environments support backend services effectively.

Starting with this multidisciplinary team enables you to address the technical, operational, and user experience aspects of your CDE deployment.

TIP

Begin small, focus on iterative improvements, and collaborate closely with developers and end users to gather feedback and refine the platform before rolling it out more broadly. Regular feedback sessions reveal pain points and identify opportunities to optimize the experience.

Making Accommodations for Midsize to Enterprise-Size Teams

Rolling out CDEs to larger teams requires a thoughtful onboarding strategy to drive adoption. Start by clearly explaining why you're transitioning to a CDE. Then provide live training sessions and accessible documentation to help developers get started.


TIP

Keep the experience familiar by ensuring that the CDE uses tools that developers are already comfortable with. Start small with dry runs for focus groups or specific teams and expand gradually. Role-based access control (RBAC) can simplify the experience by giving users access to only the templates they need, keeping the interface clean and straightforward.

Seeing How CDEs Work with Your Existing Tools and Workflows

Assuming you chose the right CDE for your organization, implementation should proceed smoothly. CDEs are designed to integrate with the tools and workflows your team already relies on. Platforms like Coder, which are cloud, integrated development environment (IDE), and language agnostic, give developers the freedom to use tools they already know, such as JetBrains IDEs or Visual Studio Code. This flexibility minimizes disruption during the transition. Modern CDEs also integrate seamlessly with version control systems such as GitHub or GitLab and authentication providers via OpenID Connect (OIDC).

Instead of replacing continuous integration/continuous deployment (CI/CD) workflows, CDEs plug directly into them. Instead of processes being triggered from local development environments on a developer's laptop, they're initiated from the CDE itself. This keeps workflows consistent while moving the activity to a more secure and scalable environment.

Smoothing the Transition


TIP

Here are a few suggestions for accelerating adoption, minimizing latency, and controlling cloud costs as you implement your new CDE:

- Demonstrate quick wins, such as faster onboarding and seamless project switching to build momentum.
- Share an informal internal case study showcasing how a trusted team uses the CDE to further encourage adoption.
- For distributed teams, ensure a multiregion architecture to minimize latency and connectivity concerns.
- Implement resource quotas and monitor usage during the transition to prevent unexpected spikes in cloud usage and costs.

Scaling Up over Time

As you add users, projects, and workspace templates to serve different teams and use cases, take the following steps to optimize the developer experience and productivity:

- Monitor the tools and IDEs developers are using in their workspaces, and conduct periodic developer experience surveys to measure satisfaction and identify areas for improvement.
- Host internal lunch-and-learn sessions to demonstrate the benefits of CDEs to nonusers and encourage broader adoption.
- Measure workspace provisioning times to identify bottlenecks and use workspace scheduling to ensure environments are ready when developers need them. Observability tools such Grafana and Prometheus can help track usage patterns and optimize resources.
- Ensure high-availability deployments for uninterrupted service as the platform scales.

IN THIS CHAPTER

- Recognizing the many benefits of a CDE
- Streamlining and accelerating software development
- Optimizing resources while enhancing security
- Working from anywhere at any time on any machine
- Using the tools you know and love more efficiently than ever

Chapter 6
Ten Ways to Win with a CDE

Cloud development environments (CDEs) help solve many of the challenges teams face, including inconsistent environments, time-consuming processes, and security risks. Whether your team is small or part of a large enterprise, adopting a CDE can boost productivity and improve the developer experience.

This chapter highlights ten practical ways CDEs can make a difference, from faster onboarding to better resource optimization. These examples highlight why more teams are embracing CDEs to improve productivity, security, and collaboration.

Faster Onboarding

Traditional developer onboarding can be slow and frustrating, with new hires spending days or weeks configuring local environments before writing their first line of code. With CDEs, onboarding becomes nearly instantaneous. Developers receive access to

preconfigured, cloud-hosted workspaces that include all the necessary tools, libraries, and dependencies for their projects.

Faster onboarding eliminates the common headaches of dependency conflicts, mismatched configurations, or waiting for access approvals. Developers can start contributing code on day one, accelerating project timelines and improving first impressions of the team. Additionally, CDEs simplify onboarding for experienced developers transitioning between projects, enabling them to quickly switch between pre-built environments without disruption.

Consistency and Compatibility across All Machines

One of the most common pain points for development teams is the challenge of ensuring consistency and compatibility across different machines, given that differences in operating systems, library versions, or untracked changes can lead to unexpected issues. CDEs eliminate this variability by centralizing environment configurations in the cloud.

Every developer on the team works from the same environment, ensuring that builds, tests, and deployments behave consistently across all stages of development. When updates or changes are required, they can be rolled out centrally to all environments, avoiding drift and reducing troubleshooting time. This consistency fosters better collaboration and reduces the risk of errors in production.

Enhanced Security

Traditional development practices can expose sensitive code and data to risks such as lost laptops or stolen credentials. CDEs mitigate these threats by centralizing code and data in the cloud, where developers access them securely without storing anything locally. For regulated industries, air-gapped deployments provide an added layer of compliance and protection.

Simplified Management

Maintaining local development environments is often a time sink for developers and platform teams. From managing updates to troubleshooting errors, these tasks distract from actual development work.

CDEs significantly simplify maintenance by shifting responsibility to a central control plane. Platform teams can manage updates, patch vulnerabilities, and roll out new tools across all developer workspaces in a single operation. Developers no longer need to handle configuration changes themselves, freeing them to focus on coding. This centralized model also reduces the time it takes to apply security fixes, ensuring a more secure and efficient development environment.

Remote Access

Remote work has made flexibility essential for developers. CDEs allow access to cloud-hosted workspaces from any device, whether a laptop, thin client, or tablet, without performance loss. This ensures productivity from anywhere while reducing reliance on specific hardware.

Resource Optimization

Traditional development environments often lead to inefficient resource usage. Developers are frequently assigned dedicated virtual machines (VMs) that run 24/7, even when they're not in use. This results in significant overprovisioning and wasted infrastructure costs.

CDEs solve this problem through smarter resource allocation. Multiple developer workspaces can run on a single VM or Kubernetes cluster, maximizing utilization without sacrificing performance. Features such as autostop and autostart further reduce costs by shutting down inactive environments. Resource quotas also enable platform teams to control infrastructure usage, preventing runaway costs and ensuring efficient use of available resources.

Seamless Integration

Adopting a new platform often introduces the risk of disruption, but modern CDEs are designed to integrate seamlessly with existing tools and workflows. Developers can continue using their preferred integrated development environments (IDEs), such as JetBrains IDEs or Visual Studio Code, while the CDE handles environment provisioning and management in the background.

CDEs also connect effortlessly with version control systems including Bitbucket, GitHub, and GitLab, as well as continuous integration/continuous deployment (CI/CD) pipelines, monitoring tools, and authentication providers. Organizations can build on their existing investments instead of overhauling their toolchains. Integration capabilities ensure that teams can adopt CDEs without disrupting productivity or requiring extensive retraining.

Faster Feedback Loops

Development speed often depends on how quickly teams can test and iterate. CDEs accelerate feedback loops by using powerful cloud infrastructure for faster builds and tests, giving developers quick insights into their changes. Centralized environments also cut down on troubleshooting time, helping teams maintain steady progress.

Improved Governance

For enterprises, governance and compliance are critical priorities that often complicate development workflows. CDEs simplify governance by providing centralized control over environments, tools, and access permissions.

Platform teams can enforce policies across all workspaces, ensuring that developers use only approved tools and libraries. Detailed logging and auditing capabilities provide visibility into how environments are being used, enabling teams to detect and respond to issues quickly. This centralized model not only improves compliance but also simplifies the management of complex enterprise requirements at scale.

Future-Proof Development

The technology landscape is constantly evolving, and organizations need platforms that can adapt to future requirements. CDEs are inherently future-proof because they're designed to integrate with modern infrastructure and workflows while remaining flexible enough to evolve with emerging trends.

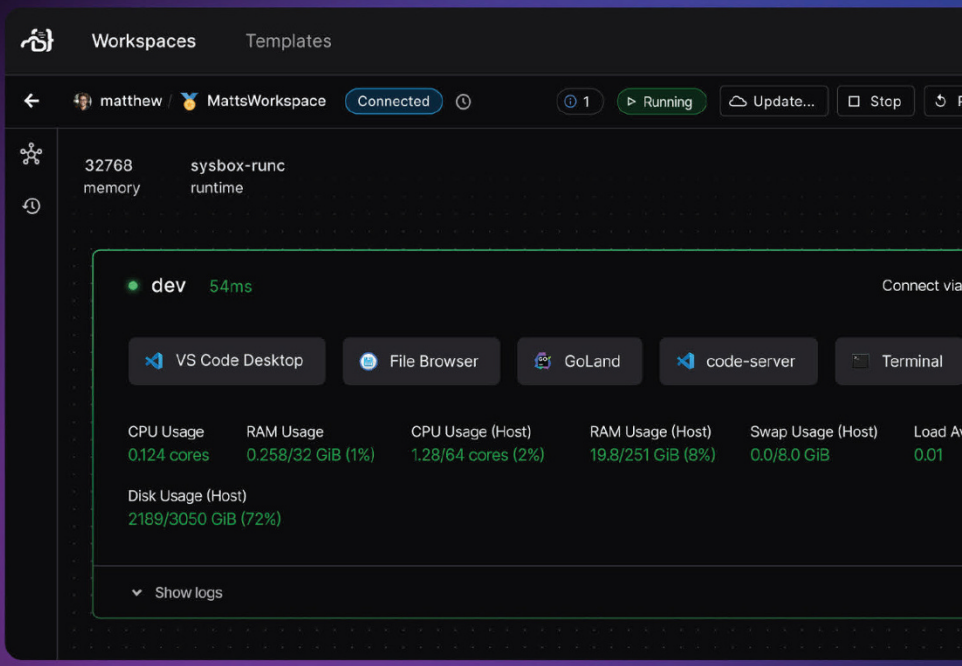
By embracing CDEs, organizations can position themselves to take advantage of new tools, frameworks, and methodologies without being constrained by legacy systems. Whether adopting artificial intelligence (AI)–driven development tools, exploring new languages, or scaling to support globally distributed teams, CDEs provide a foundation for long-term innovation and growth.

Coder | CLOUD DEVELOPMENT ENVIRONMENTS

Onboard developers to new projects in minutes (not weeks)

Self-host pre-configured development environments on your infrastructure, ready for developer use on day one.

Get started: www.coder.com



Open Source | Self-hosted | For any IDE

Migrate your development environment to the cloud

Cloud development environments (CDEs) are revolutionizing software development. Instead of juggling complicated local tools or dealing with overprovisioned virtual machines, CDEs provide a centralized space where developers can work from anywhere on any machine using all their favorite tools. It's like having your entire workspace in your back pocket — one that's faster, better equipped, easier to use, more secure, and more cost effective. In this book, you discover how to unleash the power of CDEs.

Inside...

- Onboard new developers in minutes
- Switch projects with a tap of a link
- Make developers happier and more productive
- Scale resources for builds and tests instantly
- Centralize management and governance
- Optimize resource management
- Reduce costs

Matt Vollmer is a product marketing manager specializing in developer tooling. With a background in design, development, and product, he has worked in high-growth startups, helping teams build and scale software for modern engineering challenges. He currently works at Coder, focusing on cloud development environments.

ISBN: 978-1-394-35801-4
Not For Resale

for dummies®
A Wiley Brand

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley’s ebook EULA.