What is continuous delivery?

Build, test and deploy code changes quickly for ongoing software delivery

How does continuous delivery work?

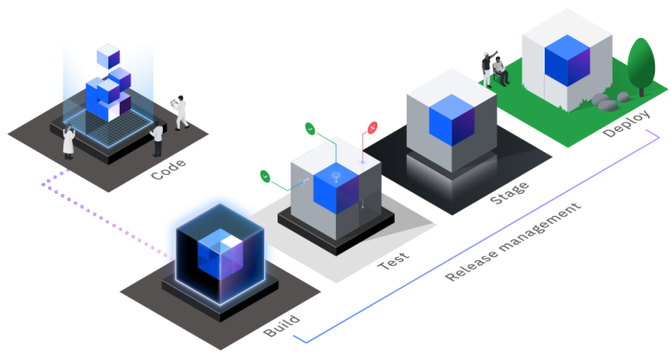
Components of continuous delivery

Trends in continuous delivery

Continuous delivery vs. deployment

How does continuous delivery work?

Continuous delivery is an automated process to get code into production quickly. It's designed to provide customers with fast access to the latest software updates so they can take advantage of new features quickly.



Stages in a continuous delivery pipeline

Continuous delivery pipelines

The continuous delivery pipeline is the core component of a continuous delivery process. The pipeline starts with code builds, then proceeds through the quality assurance, testing, staging and production environments to automate the flow of IT delivery tasks.

The goal is to run pipeline jobs in a repeatable way with minimal human intervention to release apps into production at any time. A continuous delivery pipeline works across any combination of on-premises, cloud and mainframe applications.

A starting point for DevOps

Many organizations use continuous delivery as the first step in adopting a DevOps approach. Most of the tooling and processes that make up the core of DevOps culture also support critical elements of continuous delivery, such as continuous integration for code builds and continuous testing for automated testing.

Tutorials

Adopt DevOps using continuous delivery →

Configure pipelines on Kubernetes →

Components of continuous delivery

Code moves through continuous delivery pipelines in distinct phases from build to production. While continuous delivery supports a variety of practices and tools, there is no one-size-fits-all approach. Those that adopt DevOps generally use the same automated processes in all environments to improve efficiency and reduce risks.

Processes in the pipeline

Continuous integration

A software development process where developers integrate their code frequently—at least once a day—into builds in order to identify integration issues earlier, when they are easier to fix.

→

Continuous testing

The process of incorporating automated feedback at different stages of the software development lifecycle to support better speed and efficiency when managing deployments.

→

Continuous release

The ability to replace manual coordination with templates to automate release processes and collect approvals. It works with a variety of cloud or on-premises software development tools.

→

Continuous deployment

A strategy in software development where code changes to an application are released automatically into the production environment. This automation is driven by a series of rigorous predefined tests.

→

Core principles

Make every change releasable
If you use continuous delivery to make every change releasable, you must include updated user documentation, operations runbooks and information about what's changed, for auditing purposes.

Embrace trunk-based development
Continuous delivery builds on continuous integration. Avoid branches that delay integration so that every change is built, tested and deployed together for the fastest feedback.

Deliver through an automated pipeline
To successfully implement continuous delivery, you need a well-constructed, automated delivery pipeline to ensure all your code releases move into your test and production environments in a consistent way.

Automate as much as possible
Build automated processes into your software development lifecycle to create a good, reliable delivery pipeline, not only for code builds and deployments but also for the creation of new development environments.

Aim for no downtime
When you push a new function to production, you must first validate it before deploying it to the public-running application instance. This helps ensure application availability while you make frequent, continuous delivery updates.

Release at the granularity of test
If two parts of a system must be tested together, they should be released together so you know that the parts of your system are compatible. Release automation tools are good at coordinating this kind of delivery. Alternatively, fully decouple.

Learn more about automated testing

→

Tools for continuous delivery

Repeatable templates
Use repeatable templates for build and deployment automation. If a template works for one of your apps, it should work for others. Don't let every team struggle with configuration themselves.

Toolchains
Use toolchains to build, deploy and manage your applications. These integrated sets of tools can make development and operations tasks repeatable and manageable. They can include cloud services, open source tools and third-party tools.

Build your own toolchain →

Pipeline
Define stages in a pipeline to automatically build when you push changes to a linked repository and then deploy to one or more environments. You can also incorporate builds and deployments into your toolchains to connect builds to other tools.

Collaborative tools
Find tools for team collaboration to support planning, source-code management and testing processes. This allows team members to work with more speed and visibility into processes.

Instrumentation
Use instrumentation that exposes where your team is moving quickly and where things bog down, providing a guide to becoming "more continuous."

Trends in continuous delivery

The use of continuous delivery is likely to expand rapidly in the next few years as more organizations realize its value. Software updates can be delivered with greater frequency and fewer errors, while accelerated customer feedback cycles help identify new features that customers care about.

Rapid convergence of tools

You can expect the various components of continuous delivery to converge into integrated suites over time, making the entire process easier to manage. Vendors will need to react quickly to this rapidly changing market by proving they can support expansive tool and data integration. Also look for more integrated reporting in the tool suites.

AI and machine learning

AI, machine learning and other automation technologies will become key components of a continuous delivery pipeline. These technologies may initially be used in the release automation process, where you can set up rules and incorporate data from different sources to determine release readiness, or in the deployment process to confirm deployments.

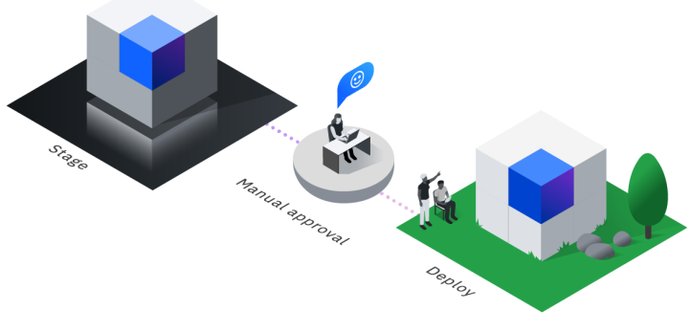
AI's role in continuous delivery

→

Continuous delivery vs. deployment

While "continuous deployment" and "continuous delivery" may sound like the same thing, they are actually two different approaches to frequent release. Both may be referred to as "CD," which can add to the confusion. Here are some examples of the differences between the two.

What you get with continuous delivery

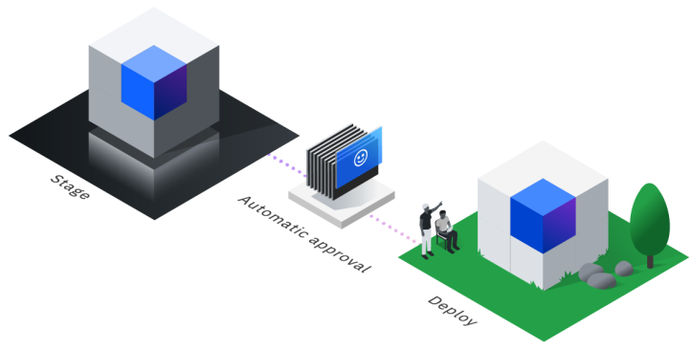


Manual approval is required to deploy

Deploy to production-like test environments
New builds performed in a continuous delivery solution are automatically deployed into a quality-assurance testing environment. Automatic tests are run for any number of errors and inconsistencies; then builds are readied for deployment.

Push builds to production manually
After the code passes all tests, continuous delivery requires some sort of human intervention to approve deployments for production. The deployment itself is then performed by automation. This manual step is common in heavily regulated industries.

The added value of continuous deployment



Automate the process through deployment

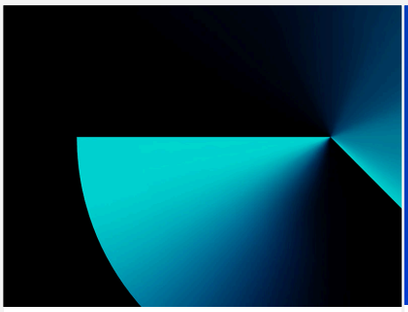
No more manual deployment approvals
Continuous deployment takes automation a step further. The tests and developers are considered trustworthy enough that an approval for production release is not required. If the code passes all tests, deployment automatically proceeds.

A fully realized pipeline
Continuous deployment is the natural outcome of continuous delivery done well. Eventually, manual approvals bring little or no value and merely slow things down. They are eliminated and continuous delivery becomes continuous deployment.

Look in depth at deployment automation

→

Learn more about it

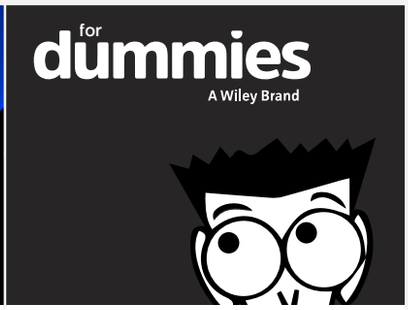


Learn

DevOps: A complete guide

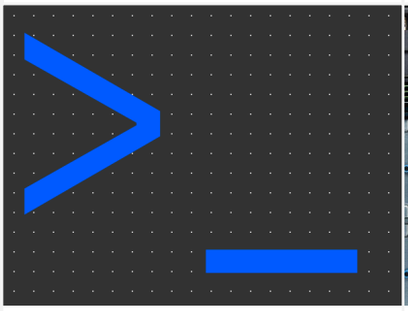
Learn

Continuous integration



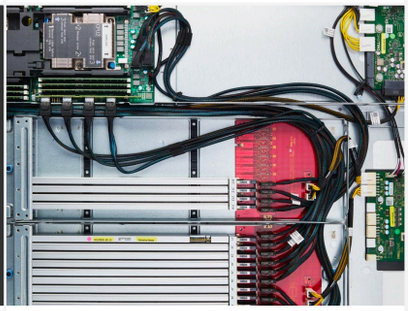
eBook

Application Release and Deployment for Dummies



Article

Best practices for implementing CI/CD



Training

Explore continuous delivery tutorials

IBM solutions

Release automation tools

Combine continuous delivery and deployment for application release.

Toolchains for app delivery

Embrace DevOps by using toolchains that support your app delivery tasks.

Continuous software testing

Get test results back to development faster and reduce risks and costs.

View all IBM continuous delivery solutions

IBM

Discover

Marketplace

Redbooks

Services

Industries

IBM Research

Case stidues

Demos

Financing

IBM Business insights

Connect with us

Support

Find a sales rep

Find a business aprtner

Contact IBM

Information for:

Developers

Business partners

Federal and state contracts

About IBM

Careers

Events

Latest news

Investor relations

Diversity and inclusion

Corporate responsibility

About IBM

United States — English

Contact IBM Privacy Terms of use Accessibility Feedback Cookie preferences