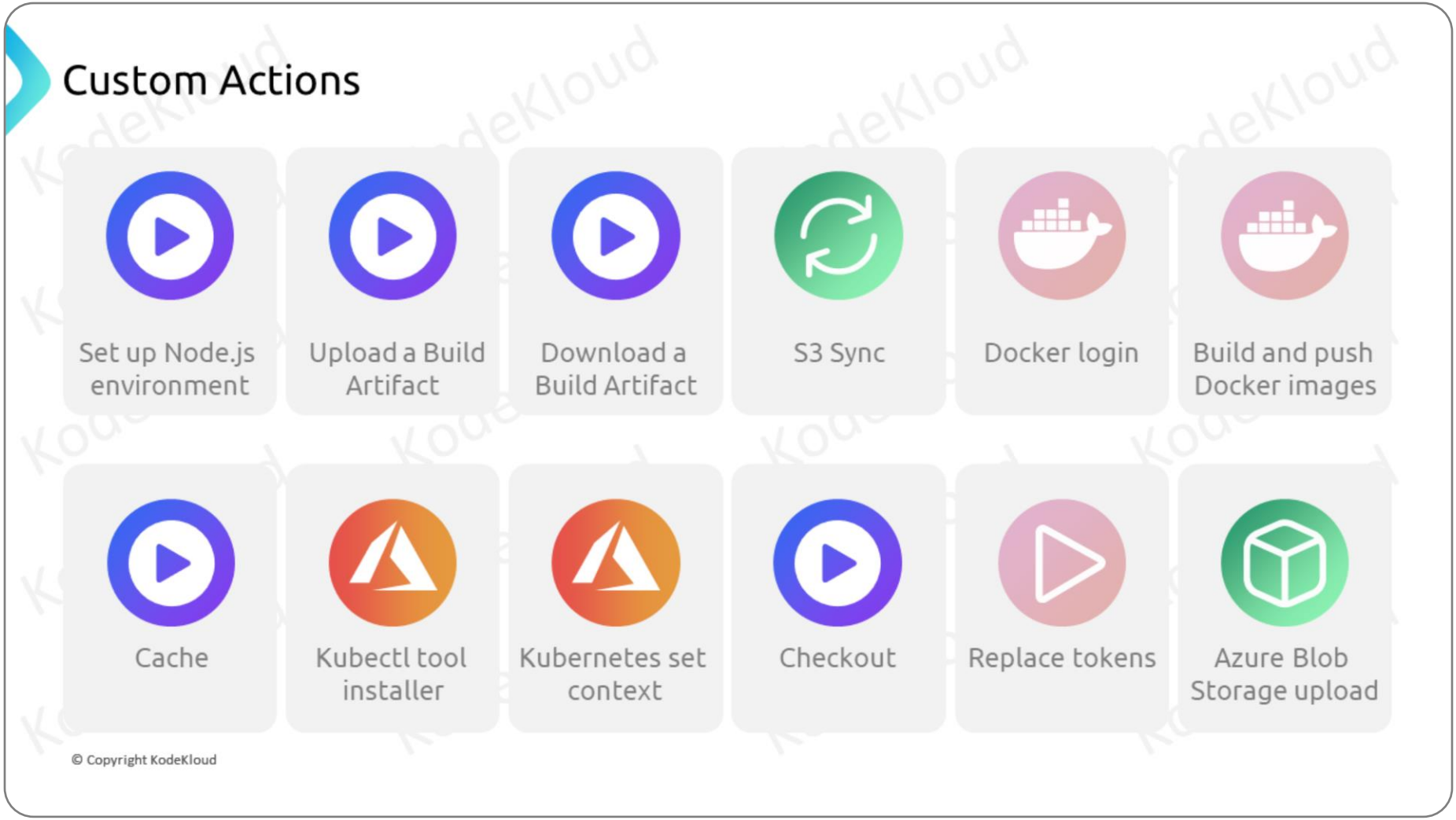
KodeKloud

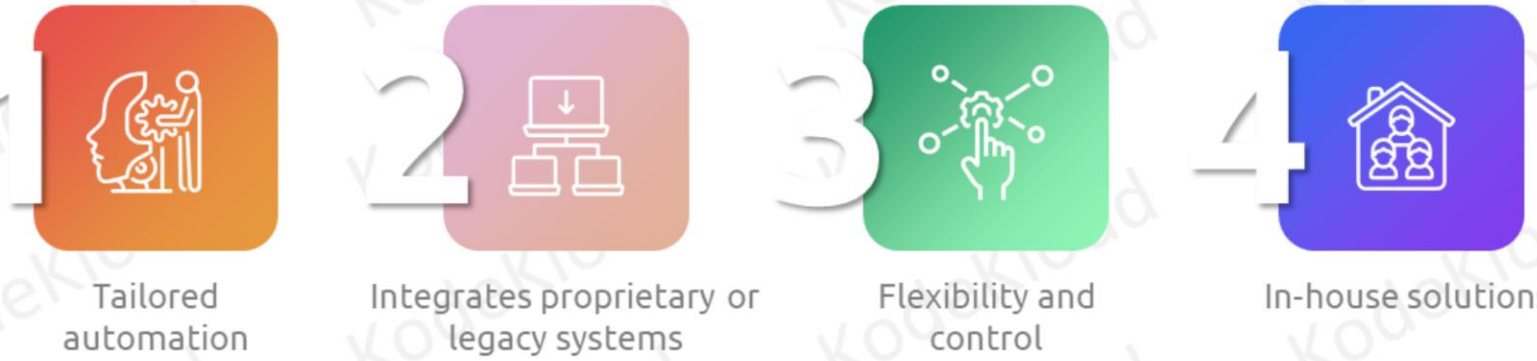
Follow us on https://kodekloud.com/ to learn more about us.

What Are Custom Actions?



Up until now, we've integrated numerous GitHub Actions from the community into our workflows. These actions have helped us with tasks like setting up runtimes, transferring artifacts, creating and publishing Docker images, syncing reports with S3 storage, and deploying to Kubernetes clusters.

Custom Actions



You may be wondering, why would we even require custom actions?

While community-contributed GitHub Actions can be incredibly useful and cover a wide range of common use cases, there are several reasons why you might still have a need for custom GitHub Actions.

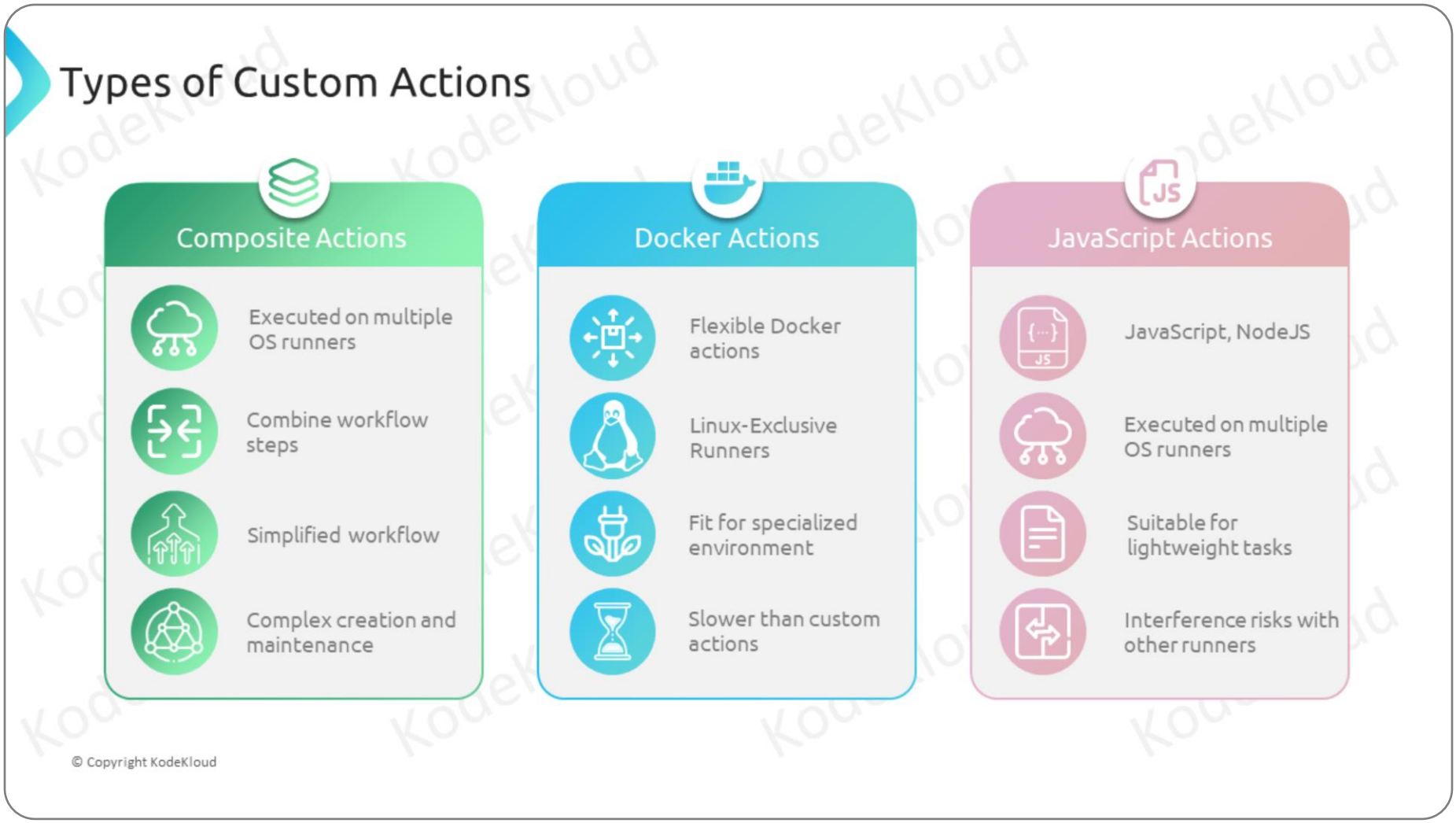
1. Your project may have unique or specific requirements that are not fully met by existing community actions. Custom actions allow you to tailor automation to your project's exact needs.

2. If your project relies on proprietary or legacy systems that are not supported by community actions, creating custom actions becomes necessary to integrate these systems into your workflow.

3. For complex workflows that involve multiple steps, conditional logic, or complicated dependencies, custom actions offer greater flexibility and control. You can create actions that orchestrate these workflows effectively.

4. Some organizations have strict security, compliance requirements and prefer to have in-house development for critical parts of their CI/CD pipelines to have full control and avoid external dependencies. Custom actions enable you to build these in-house solutions.

to summarize, while community actions are a valuable resource and can cover many common use cases, custom GitHub Actions provide the flexibility, control, and adaptability required to address specific and unique needs that may arise in your projects. They complement community actions by allowing you to extend and customize your automation workflows as necessary.



You can create actions by writing custom code that can interact with publicly available third-party API and services. For example, an action can publish npm modules, send SMS alerts when urgent issues are created, or deploy production code.

There are three main types of GitHub custom actions: Composite, Docker and Javascript actions

Composite actions: allow you to combine multiple workflow steps within one action.
This action can be executed on any Linux, macOS, Windows runners

The benefits of Composite actions is it can be used to simplify your workflows and make them more reusable. The challenges are, it can be more complex to create and maintain than other types of actions, because you need to think about how the different steps in the composite action will interact with each other.

Docker actions: run in a Docker container, which provides a clean and isolated environment for running your code. Docker is an ideal option because you can customize the operating system and tools. But you need to have Docker knowledge to build and maintain this action.
Docker container actions can only execute on runners with a Linux operating system.
Docker container actions are a good choice for tasks that require a specific environment
This action uses a Dockerfile to build and run the container and because of the latency, Docker container actions are slower than other custom actions.

Javascript actions: are written using Javascript and can use Nodejs and all of its dependencies.
JavaScript actions can run directly on a runner machine and executes faster than a Docker container action.
This action can be executed on any Linux, macOS, Windows runners
JavaScript actions are lightweight and do not require a lot of resources to run. This makes them ideal for tasks that are quick and simple.
JavaScript actions are Not as isolated as Docker container actions as they run directly on a runner machine, so they could potentially interfere with other actions running on the same runner.

KodeKloud

Follow us on https://kodekloud.com/ to learn more about us.