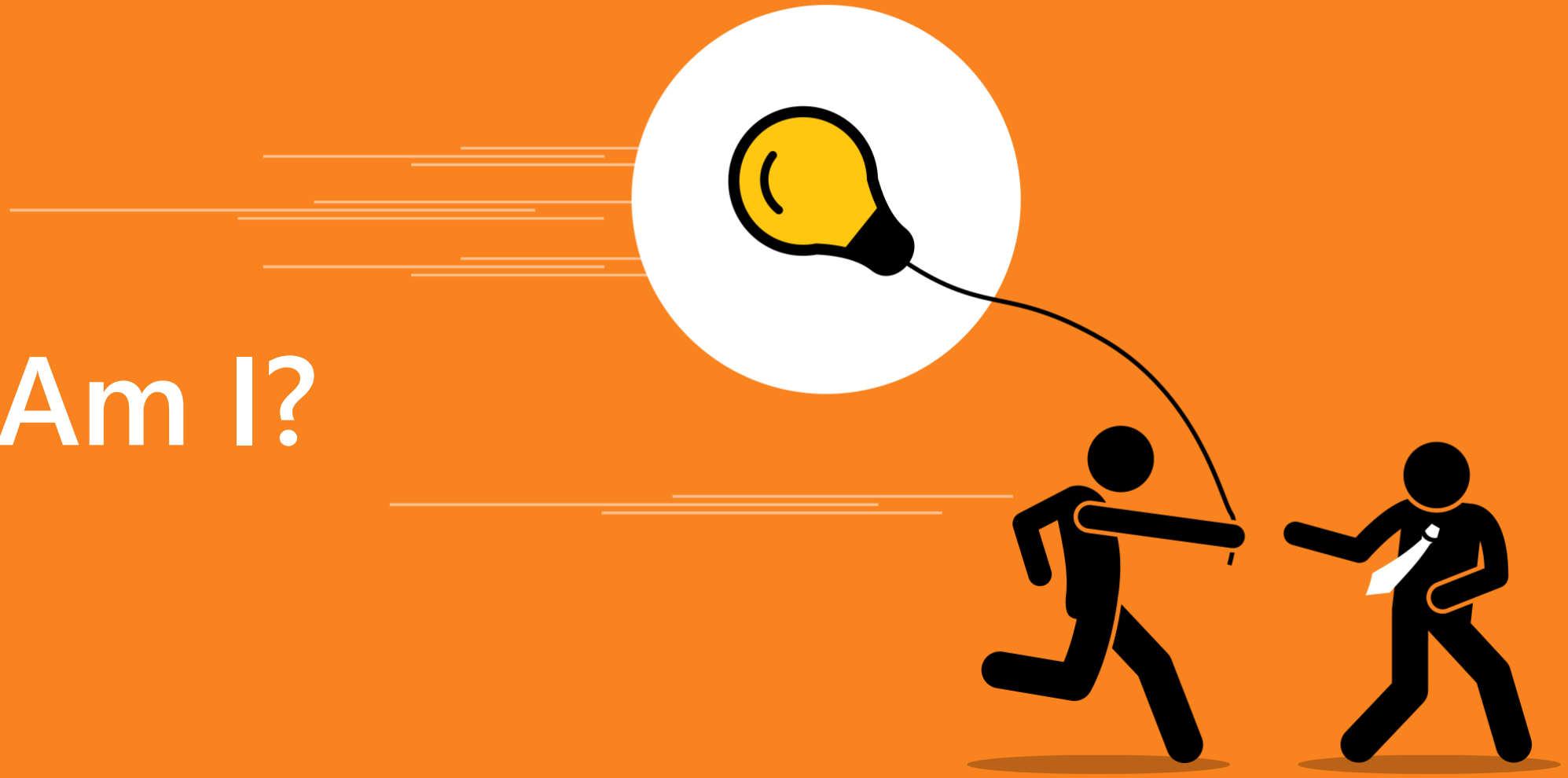
integrant

Who Am I?



integrant

GitHub Actions

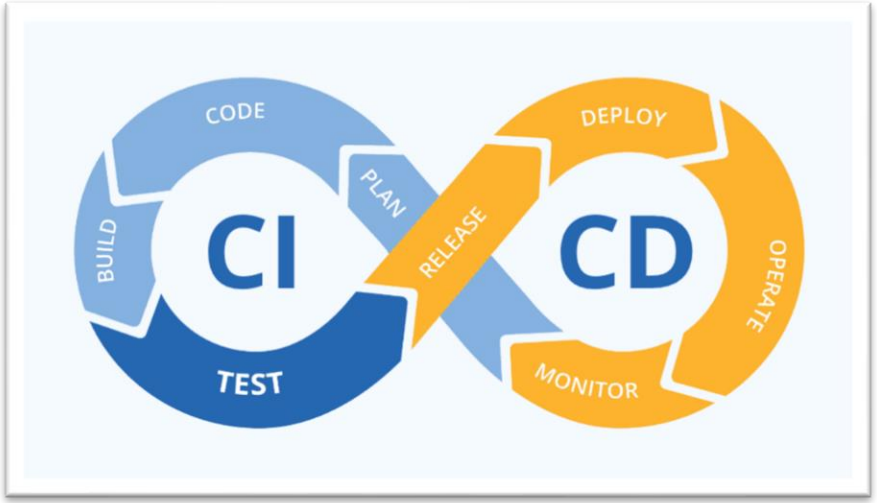
AGENDA

1. Introduction
2. Components of GitHub Actions
3. Why having a GitHub-native CI/CD tool is helpful?
4. Use Cases of GitHub Actions
5. Github Actions VS Jenkins
6. Demo

Introduction

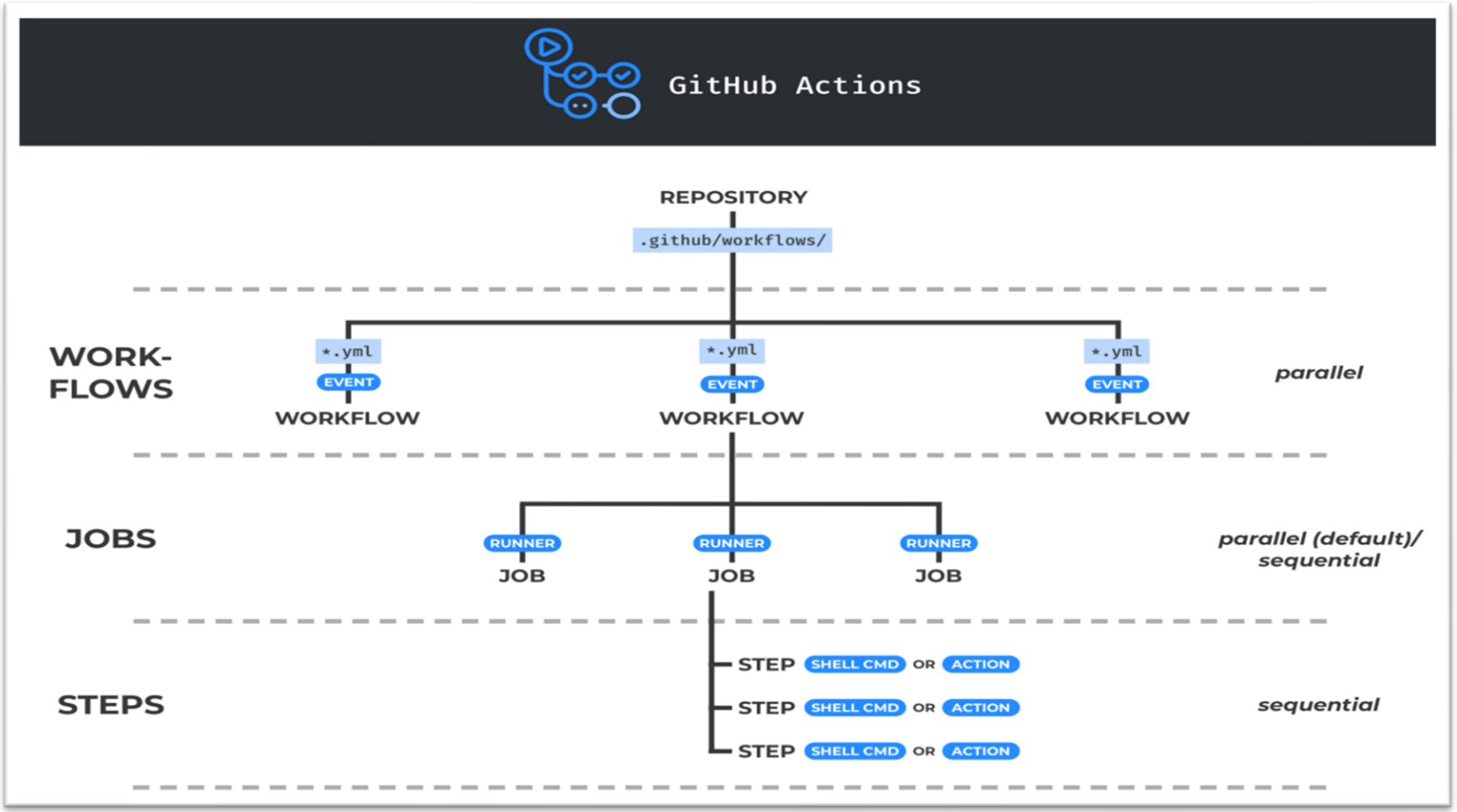
GitHub Actions

- GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform that allows you to automate your build, test, and deployment pipeline. You can create workflows that build and test every pull request to your repository or deploy merged pull requests to production.
- It was launched in 2018 as to help developers automate their workflows all within GitHub.
- GitHub Actions are event-driven, meaning that you can run a series of commands after a specified event has occurred.
- For example, every time someone creates a pull request for a repository, you can automatically run a command that executes a software testing script.



Components of GitHub Actions

- Workflows
- Events
- Jobs
- Steps
- Actions
- Runners



Components of GitHub Actions Con't

What are workflows?

- The workflow is an **automated procedure** which are like pipelines.
- **.github/workflows** directory in a repository and .**yaml** syntax.
- They are made up of one or more jobs and can be scheduled or triggered by an event.
- Listen for a particular action, then run pre-existing actions or shell scripts

```yaml
name: learn-github-actions
on: [push]
jobs:
  check-bats-version:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-node@v2
        with:
          node-version: '14'
      - run: npm install -g bats
      - run: bats -v
```

Components of GitHub Actions Con't

What are workflows?

- Workflow files glue together
- GitHub Actions also supports "**matrix builds**" which enables you to simultaneously test builds across multiple operating systems and runtime versions.
- Actions runs in VMs(Linux, Windows, Mac) or Docker Container on Linux VM
- Secret Store with each repository or organization.

```
runs-on: ${{ matrix.os }}
strategy:
  matrix:
    os: [ubuntu-16.04, ubuntu-18.04]
    node: [6, 8, 10]
steps:
  - uses: actions/setup-node@v1
    with:
      node-version: ${{ matrix.node }}
```

Components of GitHub Actions Con't

What are Events?

- An event is a specified activity that triggers a workflow. For example, activity can originate from GitHub when someone pushes a commit to a repository or when an issue or pull request is created.
- Scheduled events or Manually triggered.
- You can also use the repository dispatch webhook to trigger a workflow when an external event occurs.

What kind of Events/Triggers?

- Push
- Pull Request
- Create tag
- Create branch
- Issue labeled applied
- Issue is opened
- Make private repo to public
- Issue comment created

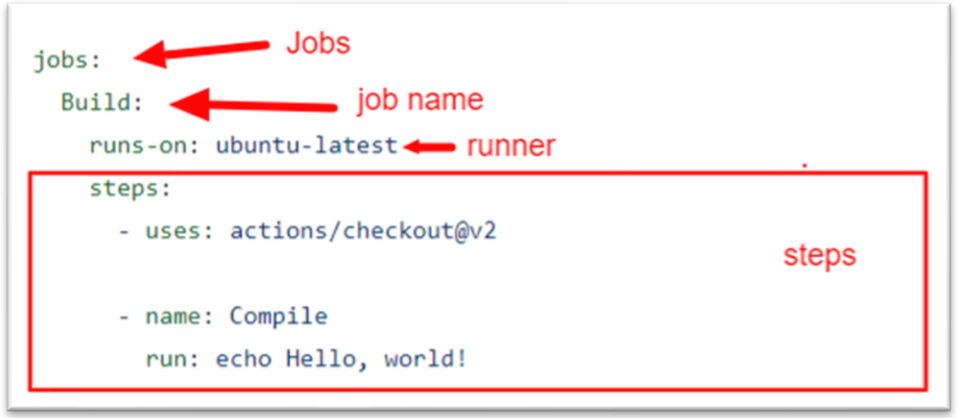
```
on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]
  workflow_dispatch:
```

Events

Components of GitHub Actions Con't

What are Jobs?

- A job is a set of steps that execute on the same runner.
- By default, a workflow with multiple jobs will run those jobs in parallel
- runs-on syntax at the beginning of the job and it's mandatory.
- Every job has its runs-on because github action engine needs to know where to basically run your workflow



Components of GitHub Actions Con't

What are Steps?

- A step is an individual task that can run commands in a job.
- A step can be either an action or a shell command.
- All steps in a job execute on the same runner, allowing the actions in that job to share data with each other.

What are Actions?

- Actions are standalone commands that are combined into steps to create a job
- Actions are the smallest portable building block of a workflow.

```
steps:
  # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
  - name: Step 1 - Checkout main branch from GitHub
    uses: actions/checkout@v2

  # Runs a single command using the runners shell
  - name: Step 2 - Set up JDK 1.8
    uses: actions/setup-java@v1
    with:
      java-version: 1.8

  - name: Step 3 - Have GitHub Actions Build Maven Project
    run: mvn -B package --file pom.xml

  - name: Step 4 - List the Current directory
    run: ls

  - name: Step 5 - what is the target directory?
    run: |
      cd target
      ls
```

Components of GitHub Actions Con't

What are Runners?

- A runner **is a server** that has the GitHub Actions runner application installed
- You can use a runner hosted by GitHub, or you can host your own.
- A runner listens for available jobs, runs one job at a time, and reports the progress, logs, and results back to GitHub.

Types of Runners:

- Github-hosted runner: are based on Ubuntu Linux, Microsoft Windows, and macOS, and each job in a workflow runs in a fresh virtual environment.
- Self-hosted runner: you can host your own runners If you need a different operating system or require a specific hardware configuration.

Note: one event can trigger many workflows, a workflow can contain many jobs, and a job can contain many steps.

Why having a github-native CI/CD tool is helpful?

- **Build into GitHub**
  - GitHub Actions is fully integrated into GitHub and therefore doesn’t require and external site. This means that it can be managed in the same place where your source code is present.
- **Multi-container testing**
  - Actions allow you to test multi-container setups by adding support for Docker and docker-compose files to your workflow.
  - it allows you to test your code in different operating systems as well as on container.
- **Multiple CI templates**
  - GitHub provides multiple templated for all kinds of CI configurations which make it extremely easy to get started as well as it also allows you to create your own templates.
- **Great free plan**
  - Actions are completely free for every open-source repository and include 2000 free build minutes per month for all your private repositories. If that is not enough for your needs, you can pick another plan or go to the self-hosted route.

Use Cases of GitHub Actions

Some of the most common use cases for GitHub Actions include:

- **Build, test, and deploy within the GitHub flow**
- **Automate repetitive tasks:** GitHub Actions can be used to automate an almost endless number of steps in the software development lifecycle. Whether it's the creation of a pull request, a new contributor joining your repository.
- **Easily add preferred tools and services to your project:** GitHub Actions gives you the ability to connect and integrate your preferred third-party tools and services directly into your repository.
- **Keep track of your projects:** you can use GitHub Actions to monitor application builds, measure performance, track errors and more via integrations with third-party tools.
- **Quickly review & test code on GitHub:** GitHub Actions lets you integrate any number of third-party testing tools directly into your workflow in your repo-at any step. Moreover, GitHub Actions enables multi-container testing and "matrix builds" which lets you run multiple tests on Linux, Windows, and macOS at the same time.

GitHub Actions VS Jenkins

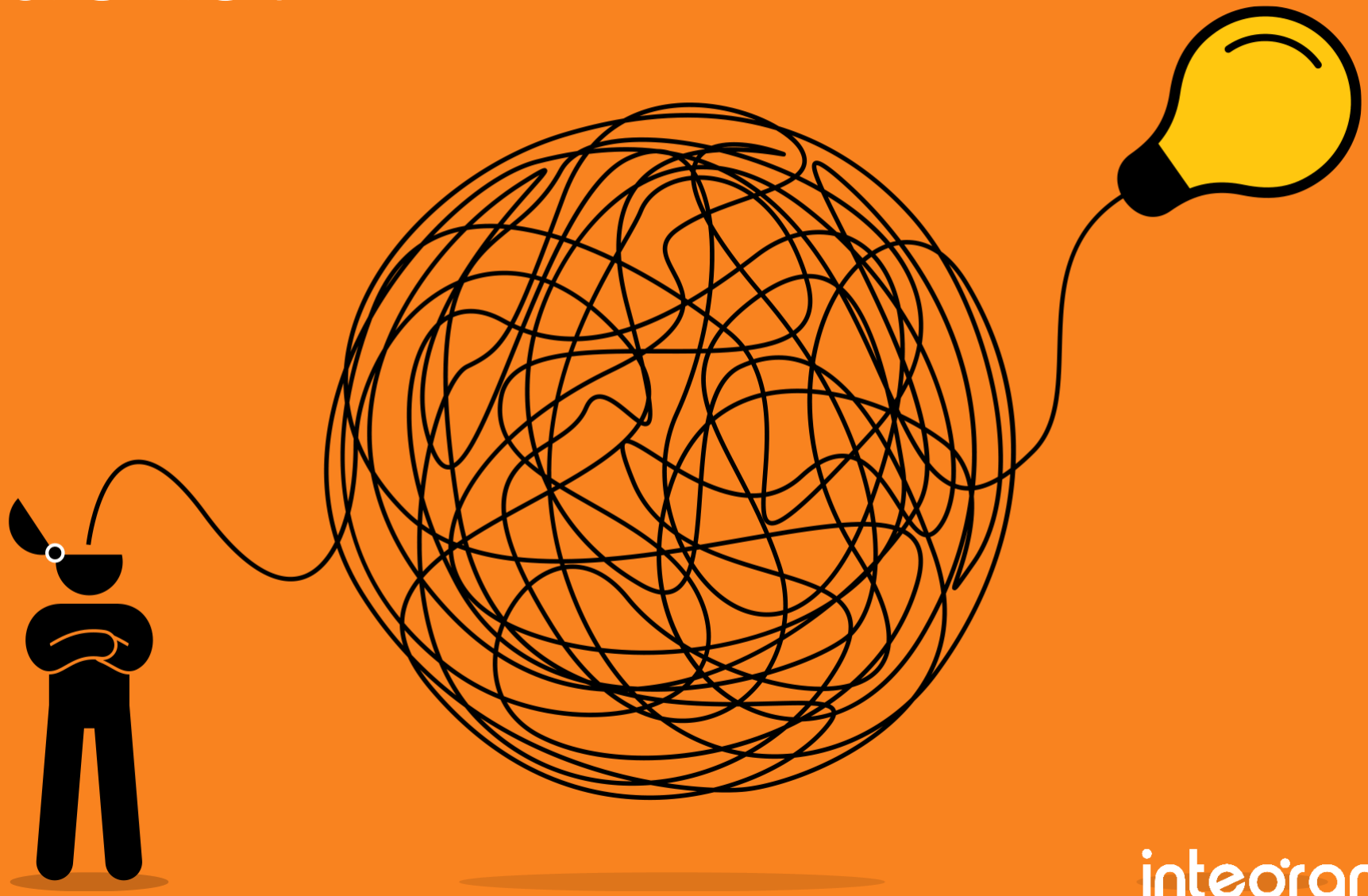
GitHub Actions VS Jenkins

	Github Actions	Jenkins
Basics	It is a fully managed service by GitHub which gives you full control over your CI/CD pipelines.	You don’t have full control of your CI/CD pipeline workflow.
Cloud	It operates in the cloud, but you can also run it on a local server. Which is called a runner.	It is a server-based application that runs in servlet containers such as Apache tomcat.
Setup	It provides a stable system with no installation required.	Jenkins server needs installation.
Pipeline	It uses jobs to put one or more steps or individual commands into groups.	It makes use of stages to run a collection of steps.

DEMO

integrant

Any Questions?



integrant

THANK YOU

integrant