

哈尔滨工业大学 计算学部

2024 年秋季学期 《开源软件开发实践》

Lab4：开源软件开发中的 DevOps

学号	姓名	联系方式
2022210939	岳翼博	19716318612

目 录

1 实验要求.....	1
2 实验内容 1 Github Actions DevOps 实践	1
3 实验内容 2 Jenkins DevOps 实践.....	4
4 小结	8

1 实验要求

1.1 实验目标:

1. 掌握开源软件开发中的基本 DevOps 流程和工具的使用。
2. 熟悉利用 Github Actions 进行 DevOps。
3. 熟悉利用 Jenkins 进行 DevOps。

1.2 实验过程:

1. Github Actions DevOps 实践:

- 1) 本地创建 Maven 项目:新建 Maven 项目并引入 Junit 依赖,导入 Lab2 中的 Debug 程序和单元测试文件。
- 2) 提交 Maven 项目到 Github 仓库。
- 3) 编写 DevOps workflow 文件: 在项目根目录下新建.github/workflows/文件夹,创建.yml 文件定义自动化测试流程。
- 4) 提交 workflow 文件到仓库,并执行自动化测试。
- 5) 再次执行自动化测试: 修改测试用例为错误结果,再次提交至仓库观察测试结果。
- 6) 选作任务: 增加 PR 的自动 merge 功能,当 PR 通过自动测试后,接受并 merge 到项目中。

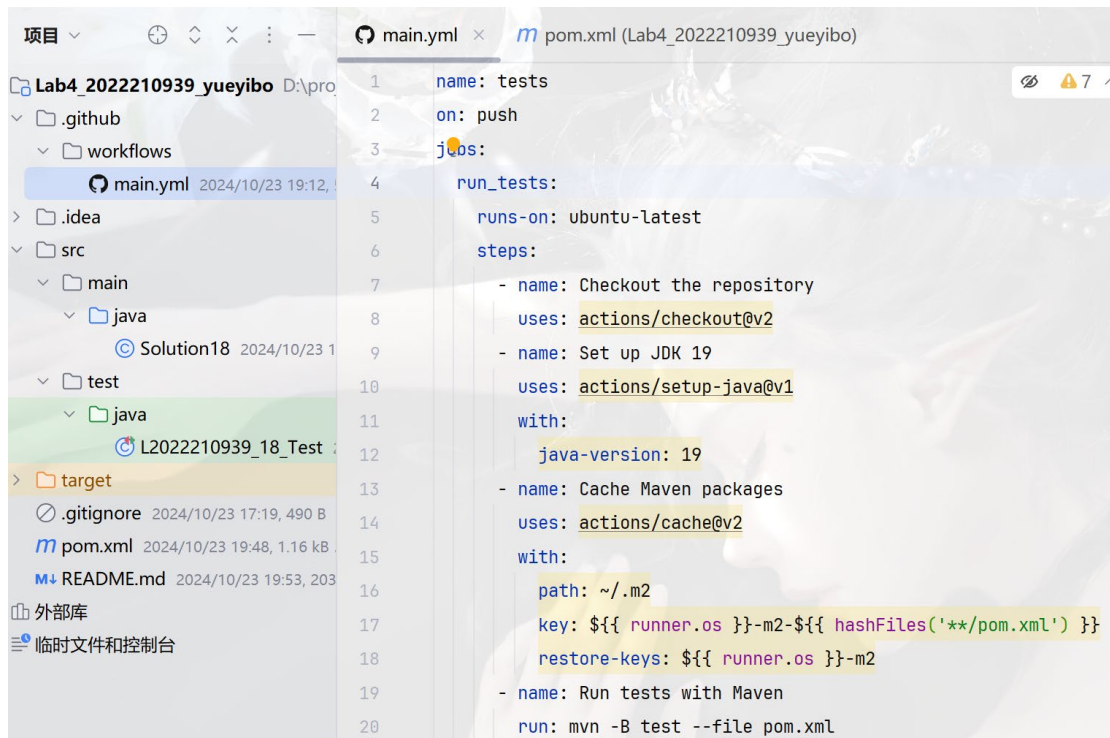
2. Jenkins DevOps 实践:

- 1) Jenkins 安装与配置: 下载安装 Jenkins,配置访问 Github 仓库的权限。
- 2) Github CLI 注意事项: 申请 Access Token 以进行认证。
- 3) 测试用 Github 仓库: 使用与 Github Actions 实践相同的仓库。
- 4) 本地创建 Maven 项目并提交。
- 5) 构建 DevOps workflow: 在 Jenkins 中创建新的构建流程,配置源码管理、构建触发器、构建步骤等。
- 6) 验证实验效果: 提交代码到指定分支,查看 Jenkins 构建结果和 Github 上的 PR 推送情况。

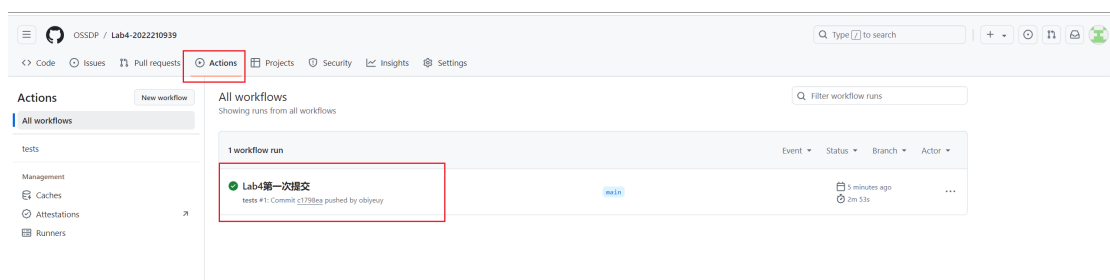
2 实验内容 1 Github Actions DevOps 实践

2.1 自动化测试成功:

- 项目名为 Lab4_2022210939_yueyibo,项目目录结构示意图如下:

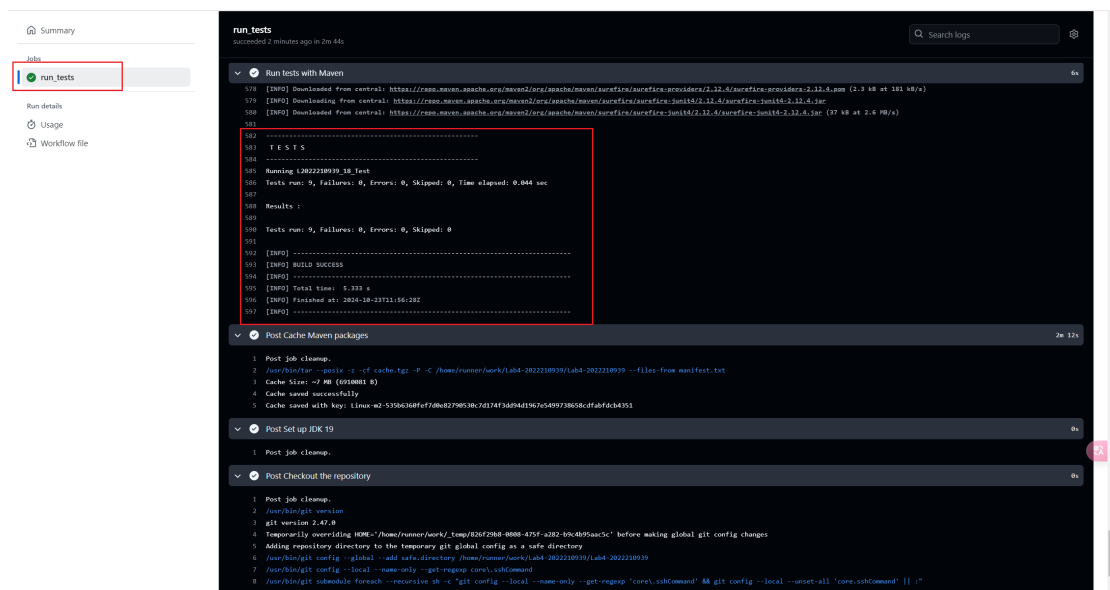


● 执行自动化测试成功界面

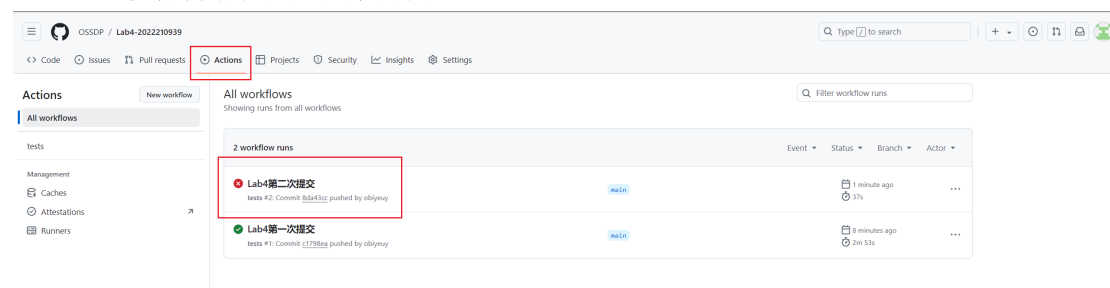


2.2 自动化测试失败:

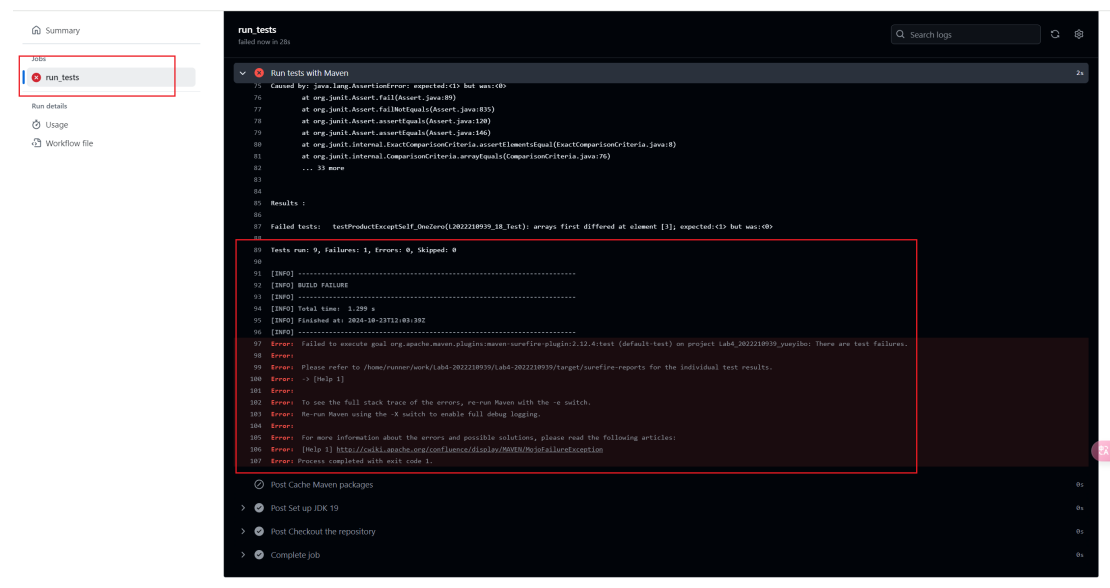
● 执行自动化测试成功具体信息界面



● 执行自动化测试失败界面



● 执行自动化测试失败界面具体信息界面



2.1 自动化测试成功:

● 给出针对步骤八的 YML 代码

```
name: Tests and Auto-merge
```

```
on:
```

```
  push:
```

```
  pull_request:
```

```
    types: [opened, synchronize, reopened]
```

```
jobs:
```

```
  run_tests:
```

```
    runs-on: ubuntu-latest
```

```
    steps:
```

```
      - name: Checkout the repository
```

```
        uses: actions/checkout@v2
```

```
      - name: Set up JDK 19
```

```
        uses: actions/setup-java@v1
```

```
        with:
```

```
          java-version: 19
```

```
- name: Cache Maven packages
  uses: actions/cache@v2
  with:
    path: ~/.m2
    key: ${{ runner.os }}-m2-${{ hashFiles('**/pom.xml') }}
    restore-keys: ${{ runner.os }}-m2

- name: Run tests with Maven
  run: mvn -B test --file pom.xml

automerge:
  needs: run_tests
  runs-on: ubuntu-latest
  if: github.event_name == 'pull_request' && github.actor != 'dependabot[bot]'
  steps:
    - name: Automerge PR
      uses: pascalgn/automerge-action@v0.15.6
      env:
        GITHUB_TOKEN: "${{ secrets.GITHUB_TOKEN }}"
        MERGE_LABELS: ""
        MERGE_METHOD: "squash"
        MERGE_COMMIT_MESSAGE: "pull-request-title"
        MERGE_REQUIRED_APPROVING_REVIEW_COUNT: "0"
        MERGE_DELETE_BRANCH: "true"
```

● 实验效果

← Tests and Auto-merge

✓ Auto PR from dev-Jenkins to main #3

Summary

Jobs

- ✓ run_tests
- ✓ automerge

Run details

Usage

Workflow file

Triggered via pull request 20 minutes ago	Status	Total duration	Artifacts
obiyeuy synchronize #2 dev	Success	3m 31s	—

merge.yml

on: pull_request

✓ run_tests 2m 40s

✓ automerge 34s

3 实验内容 2 Jenkins DevOps 实践

3.1 构建 DevOps workflow

- 新建 Item



Jenkins 查找 (CTRL+K)





Dashboard > All > 新建Item

新建Item

输入一个任务名称

测试Jenkins(开源软件开发实践实验4)

Select an item type

-  **Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
-  **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
-  **构建一个多配置项目**
适用于多配置项目,例如多环境测试,平台指定构建,等等.
-  **Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.
-  **多分支流水线**
根据一个SCM仓库中检测到的分支创建一系列流水线。
-  **文件夹**
创建一个可以嵌套存储的容器。利用它可以进行分组。视图仅仅是一个过滤器,而文件夹则是一个独立的命名空间,因此你可以有多个相同名称的内容,只要它们在不同的文件夹里即可。

确定

- 凭证管理

Jenkins Credentials Provider: Jenkins

类型

SSH Username with private key

范围 ?

Global (Jenkins, nodes, items, all child items, etc)

ID ?

描述 ?

Github SSH私钥

Username

obiyeuy

☐ Treat username as secret ?

Private Key

☒ Enter directly

Key

在下面输入新秘文

- 源码管理

源码管理

无

Git

Repositories

Repository URL

https://github.com/OSSDP/Lab4-2022210939.git

Credentials

- 无 -

+ 添加

高级

Add Repository

Branches to build

指定分支 (为空格代表any)

*/dev

Add Branch

源码浏览器

(自动)

Additional Behaviours

新增

构建触发器

POB SLM

日程表

H/3 ****

Would last have run at 2024年10月23日星期二 中国标准时间 21:56:15; would next run at 2024年10月23日星期二 中国标准时间 21:56:15.

☐ 忽略构于 post-commit

构建环境

☐ Delete workspace before build starts

☐ Use secret text(s) or file(s)

☐ Add timestamps to the Console Output

☐ Inspect build log for published build scans

☐ Terminate a build if it's stuck

☐ With Ant

Build Steps

Execute Windows batch command

命令

参阅 可用环境变量列表

npm validate

高级

Execute Windows batch command

命令

参阅 可用环境变量列表

npm test

高级

3.2 验证实验效果

- 查看 Jenkins Build 结果

7

The image shows two screenshots. The top screenshot is the Jenkins dashboard for '开源软件开发实践Lab4'. It includes a sidebar with navigation links like 'Dashboard', 'Build Now', 'Queue', 'Project', 'Git Polling Log', and 'Queue'. The main area shows '相关链接' (Related Links) with a list of build statuses: 'Last build (#1) 32 sec 之前', 'Last stable build (#1) 32 sec 之前', 'Last successful build (#1) 32 sec 之前', and 'Last completed build (#1) 32 sec 之前'. Below this is a '构建历史' (Build History) section with a table showing build numbers and status.

The bottom screenshot is a GitHub pull request interface for 'Auto PR from dev-Jenkins to main #2'. It shows a comment from 'obiyeuy' stating 'This is an auto PR from Jenkins.' and a list of commits. The right sidebar contains sections for 'Reviewers', 'Assignees', 'Labels', 'Projects', 'Milestone', 'Development', and 'Notifications'. The bottom section shows a 'Merge pull request' button and a 'Add a comment' section.

4 小结

4.1 实验过程：

本次实验通过实际操作，让我们深入理解了 DevOps 在开源软件开发中的重要性。通过使用 Github Actions 和 Jenkins 这两个流行的 DevOps 工具，实验涵盖了从代码提交、自动化测试到持续集成/持续部署（CI/CD）的完整流程。

4.2 实验结果思考：

1. 自动化的重要性：自动化测试和构建流程可以减少人为错误，确保代码质量。
2. 持续集成的价值：持续集成有助于快速发现问题，加快开发周期。

4.3 本次实验的收获:

1. 技术技能提升: 通过实践学习了 Github Actions 和 Jenkins 的配置和使用, 提升了自动化测试和 CI/CD 的技能。
2. 问题解决能力: 在实验过程中遇到的问题, 如权限配置、环境变量设置等, 锻炼了解决问题的能力。