

# GitHubActions

---

xbZhong

2025-10-24

[本页PDF](#)

## GithubAction

---

Github提供的一个**持续集成与持续部署(CI/CD)**平台

可以在**代码提交、拉取请求、发布版本**等事件发生时，**自动执行脚本或者流程**

免费额度：2000分钟/月

## 使用

需要在 `.github/workflows` 下创建一份 `yml` 文件，里面编写我们的脚本

**示例**

```

name: 部署前端项目到 GitHub Pages    # 工作流名称，会显示在 Actions 页面中

# -----
# ☑ on: 定义触发条件
# -----
on:
  push:                # 当有 push 事件时触发
    branches:          # 限制触发的分支
      - main           # 仅当推送到 main 分支时才触发
  workflow_dispatch:   # 允许手动在 GitHub 页面点击“Run workflow”执行

# -----
# ☑ jobs: 定义要执行的任务
# -----
jobs:
  build-and-deploy:    # 任务名（可以自定义）
    runs-on: ubuntu-latest # 使用最新版本的 Ubuntu 虚拟机环境

    # -----
    # ☑ steps: 定义任务中的每一步
    # -----
    steps:
      # 第 1 步: 检出代码（拉取当前仓库内容）
      - name: Checkout repository
        uses: actions/checkout@v4

      # 第 2 步: 设置 Node.js 运行环境
      - name: Setup Node.js
        uses: actions/setup-node@v4
        with:
          node-version: 18      # 指定 Node.js 版本为 18

      # 第 3 步: 安装依赖
      - name: Install dependencies
        run: npm install        # 执行命令行指令

      # 第 4 步: 构建项目
      - name: Build project
        run: npm run build      # 构建结果一般输出到 dist 目录

      # 第 5 步: 部署到 GitHub Pages
      - name: Deploy to GitHub Pages
        uses: peaceiris/actions-gh-pages@v3
        with:
          github_token: ${ secrets.GITHUB_TOKEN } # 内置的仓库访问令牌
          publish_dir: ./dist                    # 指定要发布的目录（构建产物）

      # 第 6 步（可选）: 部署完成后打印信息
      - name: Done
        run: echo "☑ 部署完成!"

```

- `name`: 工作流名称
- `on`: 选择要触发的事件类型

- `push`: 推送时触发
- `pull_request`: 别人提交合并请求时触发
- `schedule`: 定时触发
  - `branches`: 指定会触发这个工作流的**分支的事件**
- `jobs`: 顶级字段, 包含多个job, job的名字可以自定义
  - `run-on`: 运行环境, 支持三种操作系统
    - 有 `ubuntu-latest`、`windows-latest`、`macos-latest`
  - `steps`: 操作步骤, 每个step前面用 `-` 区分
    - `run`: 执行``
    - `uses`: 调用别人写好的自动化模块
    - `with`: 传递参数给 `uses` 或者 `run`
    - `|`: 多个命令一起运行
  - `env`: 设置环境变量
- `secrets`: 存储敏感信息, 在仓库设置里进行配置
- `timeout-minutes`: 设置超时时间
  - 可以在 `job` 或者 `step` 下进行设置