Magic of GitHub Actions: Automating Tasks

GitHub Actions

Seyyed Ali Mohammadiyeh (Max Base)

Tehran Lug - 27 Feb 2027

About me

Seyyed Ali Mohammadiyeh (Max Base)

Open-source Maintainer, GitHub
Software Engineer
CTO, asrez

maxbasecode@gmail.com

About me

Seyyed Ali Mohammadiyeh (Max Base)

- **GitHub**: https://github.com/basemax
- **Experience**: Over 10 years in software development and programming
- **Background**: Pure-mathematics and applied mathematics, with research experience

What is GitHub Actions?

- A CI/CD service by GitHub
- Automates workflows directly in GitHub repositories

Why Use GitHub Actions?

- Automates testing, deployment, and workflows
- Reduces manual work
- Provides seamless integration with GitHub repositories

What is CI/CD?

- CI (Continuous Integration): Merging code frequently & running tests automatically
- CD (Continuous Deployment/Delivery): Automatically deploying tested code to development/production/server

Benefits of CI/CD

- Faster development cycles
- Improved code quality
- Automatic rollback in case of failure

GitHub Actions != CI/CD

Automate your workflow from idea to production

Components of GitHub Actions

- **Workflows**: Define automation process
    - **Events**: Triggers for workflows (push, pull request, etc.)
    - **Jobs**: Tasks running in parallel or sequentially
    - **Steps**: Individual commands within a job
    - **Actions**: Pre-built or custom scripts

Workflow Example

```yaml
name: CI
on: [push]
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - name: Run a script
        run: echo "Hello, GitHub Actions!"
```

Events That Trigger Workflows

- `push` or `pull_request`
- Issues and comments
- Scheduled CRON jobs
- Manual trigger (`workflow_dispatch`)

Understanding Jobs

- A workflow can have multiple jobs
- Jobs can run in parallel or sequentially (`needs`)
- Example:

```yaml
jobs:
  test:
    runs-on: ubuntu-latest
    steps:
      - run: npm test
  deploy:
    needs: test
    runs-on: ubuntu-latest
    steps:
      - run: npm run deploy
```

Deploying with GitHub Actions

- Example deployment job:

```yaml
name: Deploy
on: push
to:
  branches: [main]
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - run: ./deploy.sh
```

Using Secrets in GitHub Actions

- Store API keys, passwords securely
- Access them using `${{ secrets.SECRET_NAME }}`
- Example:

```
jobs:
  deploy:
    steps:
      - run: echo "Deploying with ${{ secrets.API_KEY }}"
```

Matrix Strategy for Multiple Environments

```
jobs:
  test:
    strategy:
      matrix:
        os: [ubuntu-latest, windows-latest, macos-latest]
    runs-on: ${{ matrix.os }}
    steps:
      - run: echo "Running on ${{ matrix.os }}"
```

Advanced Workflow Features

- Caching dependencies
- Running workflows conditionally
- Handling workflow concurrency

Security Best Practices

- Use GitHub’s OIDC authentication for cloud providers
- Rotate secrets regularly
- Restrict permissions of GitHub tokens

Monitoring Workflow Execution

- View logs in GitHub Actions UI
- Use `job.status` for conditional steps

```yaml
jobs:
  test:
    steps:
      - run: echo "Running tests"
      - if: failure()
        run: echo "Tests failed!"
```

What are GitHub Runners?

- Machines that execute workflows
- Two types:
  - **GitHub-hosted** (Linux, macOS, Windows)
  - **Self-hosted** (Custom machine or cloud server)

How to Create a Self-hosted Runner

1. Go to GitHub repository settings
2. Navigate to `Actions` > `Runners`
3. Download and configure the runner
4. Start the runner and register it with GitHub

Example Use Cases

- Running automated tests
- Deploying applications
- Deploy Previews for Every Pull Request

???

Example Use Cases

- Automated UI Demo Creation (GIFs & Videos)
- Auto-Generate Custom Avatars for Users
- Automated Web Scraping and Data Collection
- Auto-Sync Forked Repositories
- Run Auto-Refactoring Scripts
- Auto-update Dependencies Across Multiple Repositories

Example Use Cases

- Automate Image Optimization
- Run Custom AI/ML Model Inference
- Check for Broken Links in Documentation
- Generate Graphs and Analytics
- Detect Duplicate Code and Generate Reports
- Enforce Coding Standards

Example Use Cases

- Run Code Quality Analysis on Each Pull Request
- Monitor and Report Code Vulnerabilities
- Automated Feature Flag Management
- Create or Update GitHub Pages Automatically
- Deploy to Multi-Cloud Environments
- Run Stress Tests on the Codebase

Example Use Cases

- Create a Performance Benchmarking Pipeline
- Automated Merge Conflict Detection
- Personalized Onboarding for New Contributors
- Convert Documentation to Different Formats
- Sending notifications (Slack, Email)
- Trigger a Workflow on a Specific Day or Time

Example Use Cases

- Trigger Auto-Deploys Based on Custom Labels
- Automating documentation generation
- Enforce Versioning Standards
- Automate Software Licensing Checks
- Auto-generate Release Notes
- Run Cryptocurrency or Blockchain-related Jobs

Example Use Cases

- Running security scans
- Auto release Android(apk) and iOS release

GitHub Actions != CI/CD

GitHub Actions is often viewed primarily as a CI/CD tool, It can automate all sorts of tasks throughout the software development lifecycle, from ideation to production and even beyond.

Brew Coffee by GitHub Actions

Brew a coffee as a gift once the official developers of the project make a successful, error-free commit.

Cool builds

[Using GitHub Actions to Brew Coffee]

[Hacking Bluetooth to Brew Coffee on GitHub Actions Part 1]

Real-World Examples

1. https://github.com/BaseMax/React-Auto-Build-GitHub-Actions
2. https://github.com/BaseMax/github-actions-nextjs-build-deploy
3. https://github.com/BaseMax/AndroidAutoBuildAPK
4. https://github.com/BaseMax/GitHubAction-Jekyll-SFTP-Deploy-Password
5. https://github.com/BaseMax/GitHubAction-SFTP-Deploy-Password
6. https://github.com/BaseMax/AutoInviteToOrgByIssueComment

Real-World Examples

7. https://github.com/BaseMax/AutoInviteToOrgByStar
8. https://github.com/BaseMax/github-actions-cpanel-php-ftp
9. https://github.com/BaseMax/github-actions-compile-golang
10. https://github.com/BaseMax/github-actions-compile-c
11. https://github.com/BaseMax/github-actions-update-push
12. https://github.com/BaseMax/github-actions-create-tag

Real-World Examples

13. https://github.com/BaseMax/github-actions-upload-temp-file
14. https://github.com/BaseMax/github-actions-create-release
15. https://github.com/BaseMax/github-actions-monitor-issues
16. https://github.com/BaseMax/github-actions-run-docker-compose
17. https://github.com/BaseMax/github-actions-run-dockerfile
18. https://github.com/BaseMax/github-actions-compile-rust

Real-World Examples

19. https://github.com/BaseMax/github-actions-react-deploy-tailwindcss-sftp
20. https://github.com/BaseMax/github-actions-react-deploy-linux-sftp
21. https://github.com/BaseMax/github-actions-react-build-linux-sftp
22. https://github.com/BaseMax/github-actions-file-linux-ssh-sftp

Q&A

Let's discuss! 🚀

Repository: github.com/BaseMax/github-actions-tehlug

Linkedin: linkedin.com/in/maxbase

Email: maxbasecode@gmail.com

Telegram: t.me/MAX_BASE