

# GitHub Action Project Structure

JavaScript-based GitHub Action with inputs, outputs, and the official toolkit packages.

Updated 2025-12-22

#github #action #ci-cd #automation #javascript

PNG

PDF

Copy

</> Prompt

## Project Directory

### my-action/

- action.yml Action metadata...
- src/ Source code
  - index.js Entry point
  - main.js Main action log...
  - post.js Post-run cleanu...
- utils/
  - inputs.js Parse and valid...
  - github.js GitHub API help...
- dist/ Bundled output ...
  - index.js ncc compiled bu...
- \_\_tests\_\_/ Jest tests
  - main.test.js
  - inputs.test.js
- .github/ Workflows for t...
  - workflows/
    - test.yml Run tests on PR
    - release.yml Build and tag r...
- package.json
- .gitignore
- README.md Usage documenta...
- LICENSE

## Why This Structure?

This structure uses `@vercel/ncc` to bundle all dependencies into a single `dist/index.js` file that gets committed. This eliminates the need for `node_modules` at runtime. The `action.yml` defines the public interface with inputs, outputs, and branding.

## Key Directories

**action.yml** - Defines inputs, outputs, runs configuration

**src/** - Source code with main logic and utilities

**dist/** - Bundled code (committed to repo)

**.github/workflows/** - CI for testing and releasing the action itself

## </> action.yml Structure

```
# action.yml
name: 'My Action'
description: 'Does something useful'
inputs:
  token:
    description: 'GitHub token'
    required: true
  config-path:
    description: 'Path to config file'
    default: '.github/config.yml'
outputs:
  result:
    description: 'The action result'
runs:
  using: 'node20'
  main: 'dist/index.js'
```

## > Getting Started

- `npm init -y`
- `npm install @actions/core @actions/github`
- `npm install -D @vercel/ncc jest`
- Create `action.yml` with inputs and outputs
- Write action logic in `src/main.js`
- `npx ncc build src/index.js -o dist`

## Actions Toolkit Packages

**@actions/core** - Inputs, outputs, logging, secrets

**@actions/github** - Authenticated Octokit client

**@actions/exec** - Run shell commands

**@actions/io** - File system operations

## Best Practices

- Always bundle with `ncc` and commit `dist/`
- Use semantic versioning with major version tags (v1, v2)
- Validate inputs early with clear error messages
- Use `core.setSecret()` to mask sensitive values
- Add a `post` step for cleanup if needed

## Release Strategy

Tag releases with full semver ( `v1.0.0` ) and maintain a floating major tag ( `v1` ) that points to the latest `v1.x.x` . Users reference `uses: owner/action@v1` to get updates automatically.

## When To Use This

- Automating repetitive workflow tasks
- Integrating external services into CI/CD
- Enforcing policies on PRs or commits
- Publishing packages or deploying apps
- Generating reports or notifications

## Trade-offs

**Bundle committed** - `dist/` adds to repo size but required for actions

**Node.js only** - Use Docker action for other languages

**Testing** - Hard to test GitHub context without mocking