GitHub Releases — Practical Quick-Reference

Scope. This is a hands-on, copy-and-paste quick-reference for creating, templating, automating, and consuming GitHub Releases. Works for public and private repositories (private requires authentication where noted).

1 What is a Release?

A GitHub Release ties a repository state (a Git tag) to human-readable notes and downloadable assets (binaries, installers, SBOMs, signatures). GitHub also auto-attaches source archives (`.zip` and `.tar.gz`) for the tagged commit.

2 Latest Release: One-Liners

Open the latest release page

```
xdg-open "https://github.com/<OWNER>/<REPO>/releases/latest"
# macOS: open "https://github.com/<OWNER>/<REPO>/releases/latest"
```

Download a known asset from the latest release

```
curl -L -o sbom.spdx.json \
  "https://github.com/<OWNER>/<REPO>/releases/latest/download/sbom.spdx.json"
```

Note. The `/releases/latest` route resolves to the most recent *non-draft, non-prerelease* release.

Atom/RSS feed for releases

```
https://github.com/<OWNER>/<REPO>/releases.atom
```

3 Auto-Generated Notes with `.github/release.yml`

GitHub can generate release notes from merged PRs since the previous tag and categorize them with a template.

Minimal template

```yaml
changelog:
  exclude:
    labels: [skip-changelog, ci]
    authors: [dependabot]
  categories:
    - title: Breaking Changes
      labels: [breaking-change, Semver-Minor]
    - title: Features
      labels: [feature, enhancement]
    - title: Fixes
      labels: [bug, fix]
    - title: Other Changes
      labels: ["*"]
```

- Place this at `.github/release.yml` on the default branch.
- In the "Draft a new release" UI, click *Generate release notes*; the template drives grouping.

4 Prefill the "New Release" Form via URL

Craft a link that opens the *New release* page with fields pre-populated.

Pattern

```
https://github.com/<OWNER>/<REPO>/releases/new?
  tag=<TAG>&
  target=<BRANCH_OR_SHA>&
  title=<TITLE>&
  body=<URL_ENCODED_TEXT>&
  prerelease=true|false
```

Example

```
https://github.com/acme/widgets/releases/new?
tag=v1.2.3&
target=main&
title=Release%201.2.3&
body=Highlights%3A%0A-%20New%20renderer%0A-%20Faster%20startup&
prerelease=false
```

5 Automate Releases in CI (GitHub Actions)

Create and upload a release whenever a SemVer tag is pushed.

Workflow: create on tag push

```yaml
name: Release on tag
on:
  push:
    tags:
      - "v*"
jobs:
  release:
    runs-on: ubuntu-latest
    permissions:
      contents: write   # required to create releases
    steps:
      - uses: actions/checkout@v4

      - name: Build artifacts
        run: |
          ./scripts/build.sh
          # produce files under dist/

      - name: Create GitHub Release
        uses: softprops/action-gh-release@v2
        with:
          generate_release_notes: true
          files: |
            dist/*.tar.gz
            dist/*.zip
            dist/sbom.spdx.json
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

Alternative: CLI-driven (gh)

```bash
# Create a release with notes auto-generated from commits/PRs
gh release create v1.2.3 \
  --generate-notes \
  dist/*.zip dist/*.tar.gz dist/sbom.spdx.json
# View details (assets, URLs)
gh release view v1.2.3 --json url,tagName,assets --jq '.'
```

6 REST API Essentials

Get latest release (REST)

```
curl -s -H "Accept: application/vnd.github+json" \
  https://api.github.com/repos/<OWNER>/<REPO>/releases/latest
```

List all releases (paginated)

```
curl -s -H "Accept: application/vnd.github+json" \
  "https://api.github.com/repos/<OWNER>/<REPO>/releases?per_page=50&page=1"
```

Upload an asset (REST)

```
# After creating a release, use the upload_url returned by the API:
# "upload_url":
↪  "https://uploads.github.com/repos/<OWNER>/<REPO>/releases/<id>/assets{?name,label}"

UPLOAD_URL="https://uploads.github.com/repos/<OWNER>/<REPO>/releases/<id>/assets?name=sb⌟
↪  om.spdx.json"
curl -X POST \
  -H "Authorization: Bearer $GITHUB_TOKEN" \
  -H "Content-Type: application/json" \
  --data-binary @sbom.spdx.json \
  "$UPLOAD_URL"
```

7 Consumers: Download Patterns

Pinned asset name (preferred)

```
# Stable URL when asset filename is consistent across releases
curl -L -o tool-linux-amd64.tgz \
  "https://github.com/<OWNER>/<REPO>/releases/latest/download/tool-linux-amd64.tgz"
```

Private repositories

```
# Use a token; either a classic token or a fine-grained PAT with repo contents scope
curl -L -H "Authorization: Bearer $GITHUB_TOKEN" \
  -o tool.tgz \
  "https://api.github.com/repos/<OWNER>/<REPO>/releases/assets/<ASSET_ID>"
```

8 Release Hygiene Checklist

- **Tags**: Use signed tags (`git tag -s vX.Y.Z -m "..."`) for provenance.
- **SemVer**: Reserve `vX.Y.Z` for stable; mark prereleases as `prerelease=true`.
- **Assets**: Include platform-specific binaries, SBOMs (`.spdx.json`), checksums (`.sha256`), and signatures (`.sig`).
- **Notes**: Use `.github/release.yml` and labels for clean, auto-generated sections.
- **Verification**: Add a `README` section showing how to verify checksums and signatures.

9 Troubleshooting

- **404 on `/releases/latest/download/<FILENAME>`**: The asset name does not exist in the latest release. Confirm with:
  `gh release view -json assets -jq '.assets[].name'`
- **Prerelease not resolved by "latest"**: The `/latest` route skips prereleases. Use the explicit tag URL `/tag/<TAG>` instead.
- **Private repo downloads fail**: Use authenticated API endpoints or `gh release download`; pass a token with the correct scopes.
- **Auto-notes empty or mis-grouped**: Ensure PRs are merged (not squash without PR metadata), labels match your template, and the previous tag exists.

10 Quick Copy-Paste Snippets

Verify checksum

```
curl -L -o tool.tgz \
  "https://github.com/<OWNER>/<REPO>/releases/latest/download/tool-linux-amd64.tgz"
curl -L -o tool.tgz.sha256 \
  "https://github.com/<OWNER>/<REPO>/releases/latest/download/tool-linux-amd64.tgz.sha25
  ↪  6"

sha256sum -c tool.tgz.sha256   # OK if sum matches
```

Generate checksums during build

```
for f in dist/*; do
  sha256sum "$f" > "$f.sha256"
done
```

Sign artifacts (example: minisign)

```
minisign -Sm dist/tool-linux-amd64.tgz
# publish dist/tool-linux-amd64.tgz.minisig and your public key
```

11 Release Notes Template Hints

- Label PRs consistently: `feature`, `enhancement`, `bug`, `breaking-change`.
- Exclude noise labels (`ci`, `skip-changelog`) via `.github/release.yml`.
- Add a **Highlights** section at the top for top-3 changes and upgrade notes.

12 Link References

- Latest page: `https://github.com/<OWNER>/<REPO>/releases/latest`
- Tag page: `https://github.com/<OWNER>/<REPO>/releases/tag/<TAG>`
- Atom feed: `https://github.com/<OWNER>/<REPO>/releases.atom`
- REST: `https://api.github.com/repos/<OWNER>/<REPO>/releases`

Build tip. If you see minted errors about Pygments output, compile with `latexmk -pdf -shell-escape`. Avoid using `frozencache` unless the cache has been generated on the same machine.