Release Process (3.9+)

Releases · https://namazustudios.com/docs/releases/release-process/

Starting with Elements 3.9, every build artifact, from a snapshot straight off `main` to a formal release, comes from one of three channels. This page explains what each channel is, how stable it is, and where to get its jars, Maven coordinates, and Docker images. It's written for game and plugin developers choosing what to depend on; if you're a maintainer cutting a release, see `ACTIONS.md` in the engine repository for the GitHub Actions workflow reference.

The Three Channels

Channel	Version looks like	Stability	Jars / SDK	Docker images
Snapshot (`main`)	`3.10.0-SNAPSHOT`	Unstable, untested by hand — every push at least passes the full integration test suite	Not available for normal use	Available
Release Candidate (`release/X.Y`)	`3.9.0-rc-3`	Not fully tested, potentially unstable	Maven Central + local SDK	Available
Formal Release (`release/X.Y` , tag)	`3.9.0`	Tested, stable	Maven Central + local SDK	Available

Snapshots

Every push to `main` triggers a full build, test, and publish. Snapshot jars aren't published anywhere useful for pulling into a game project: Maven Central rejects `-SNAPSHOT` uploads outright, and the copy that does land in the GitHub Packages snapshot repository requires GitHub authentication to resolve, so treat snapshot jars as unavailable. Docker images are the one artifact that *is* readily available from this channel; see Docker Images below.

Always consider a snapshot build unstable and untested, even though it's guaranteed to have passed the full integration test suite (a snapshot build that fails those tests never finishes publishing).

Release Candidates

When a release line is ready to stabilize, it gets its own `release/X.Y` branch (e.g. `release/3.9`), cut from `main`. Every version on that branch is tagged `X.Y.Z-rc-N` until the line is formally released, e.g. `3.9.0-rc-1`, then `3.9.0-rc-2`, and so on as fixes land.

Release candidates are not fully tested and should be considered potentially unstable, but unlike snapshots they're real, resolvable artifacts: published to Maven Central under the `dev.getelements.elements` group, so you can pull an RC into your Element or SDK project exactly like a formal release, just by pinning the `-rc-N` version. Docker images are published as well.

Formal Releases

Once a release candidate has been tested and is considered stable, the `-rc-N` designation is dropped (e.g. `3.9.0-rc-3` → `3.9.0`) and a formal release is tagged and published: Maven Central, Docker images, and a GitHub Release with generated notes. This is the version you should target for production.

The release branch doesn't stop there — it immediately moves on to the next patch's first release candidate (e.g. `3.9.1-rc-1`) so bug-fix RCs for that line can keep iterating without a separate step.

Maven Coordinates

Release and release-candidate artifacts share the `dev.getelements.elements` group ID on Maven Central. Point your SDK or Element project's dependency at the exact version you want:

```
<dependency>
    <groupId>dev.getelements.elements</groupId>
    <artifactId>sdk</artifactId>
    <version>3.9.0-rc-3</version> <!-- or 3.9.0 once released -->
</dependency>
```

See Packaging an Element with Maven for how these coordinates fit into an Element project.

Docker Images

Every channel, snapshot, RC, and formal release, publishes the same two images to both Docker Hub and GitHub Container Registry:

- `elementalcomputing/elements-base` / `ghcr.io/namazustudios/elements-base`
- `elementalcomputing/elements-jetty-ws` / `ghcr.io/namazustudios/elements-jetty-ws`

Each image is tagged with the exact Maven version it was built from (e.g. `3.9.0-rc-3`, `3.10.0-SNAPSHOT`), plus the full and short git commit SHA it was built from. Because a snapshot build reuses the same `-SNAPSHOT` version across many commits on `main`, that version tag is a moving target, pull it to always get the latest snapshot, or pin a commit SHA tag to get a specific snapshot build:

```
docker pull elementalcomputing/elements-jetty-ws:3.9.0-rc-3
docker pull ghcr.io/namazustudios/elements-jetty-ws:3.10.0-SNAPSHOT
```

See Common Issues with Docker if you run into trouble running these locally.

Choosing a Channel

- **Production games:** use a formal release.
- **Testing an upcoming release, or you need a fix that's already landed but not yet released:** use the latest release candidate for that line.
- **Trying out unreleased engine work, or contributing to Elements itself:** the snapshot Docker images are the only practical way to run `main` locally, since there's no resolvable snapshot jar.