Automate Releases

If your project follows a semantic versioning, it may be a good idea to automatize the steps needed to do a release. The recipe below bumps the project version, commits the changes to git and creates a new GitHub release.

For publishing a GitHub release you'll need to create a personal access tokenand add it to your project. However, we don't want to commit it, so we'll use dotenv to load it from a git-ignored .env file:

```
GH_TOKEN=ff34885...
```

Don't forget to add .env to your .gitignore.

Next, install all the necessary dependencies for this recipe:

```
npm install --save-dev conventional-recommended-bump conventional-changelog-cli conventional-github-releaser dotenv execa
```

Based on your environment, setup and preferences, your release workflow might look something like this:

```
const gulp = require('gulp');
const conventionalRecommendedBump = require('conventional-recommended-bump');
const conventionalGithubReleaser = require('conventional-github-releaser');
const execa = require('execa');
const fs = require('fs');
const { promisify } = require('util');
const dotenv = require('dotenv');
// load environment variables
const result = dotenv.config();
if (result.error) {
throw result.error;
}
// Conventional Changelog preset
const preset = 'angular';
// print output of commands into the terminal
const stdio = 'inherit';
async function bumpVersion() {
// get recommended version bump based on commits
const { releaseType } = await promisify(conventionalRecommendedBump)({ preset });
```

```
// bump version without committing and tagging
await execa('npm', ['version', releaseType, '--no-git-tag-version'], {
stdio,
});
}
async function changelog() {
await execa(
'npx',
[
'conventional-changelog',
'--preset',
preset,
'--infile',
'CHANGELOG.md',
'--same-file',
],
{ stdio }
);
}
async function commitTagPush() {
// even though we could get away with "require" in this case, we're taking the
safe route
// because "require" caches the value, so if we happen to use "require" again
somewhere else
// we wouldn't get the current value, but the value of the last time we called
"require"
const { version } = JSON.parse(await promisify(fs.readFile)('package.json'));
const commitMsg = `chore: release ${version}`;
await execa('git', ['add', '.'], { stdio });
await execa('git', ['commit', '--message', commitMsg], { stdio });
await execa('git', ['tag', `v${version}`], { stdio });
await execa('git', ['push', '--follow-tags'], { stdio });
}
function githubRelease(done) {
conventionalGithubReleaser(
{ type: 'oauth', token: process.env.GH_TOKEN },
{ preset },
done
);
}
exports.release = gulp.series(
bumpVersion,
changelog,
commitTagPush,
```

```
githubRelease
);
```