# Lucky Parking changesets 2024

by aletia trepte, parcheesime

## Understanding Changesets:

Changesets are records of changes to code, specifically code that should be released together, providing a structured way of bundling together changes made to a codebase, making it easier to manage versions, release notes, and package publishing.

Changesets are useful where there might be multiple packages being developed together. When using changesets, the 3 types you can specify are:

**Major:** this indicates incompatible API changes, or modifications to code that requires users to change their existing code in order to continue using the product/service. Used with strategic updates, depreciation, or security issues. Major changesets indicate need for caution and review, and that significant changes may impact current use of product.

**Minor:** increment the minor version when you add features or enhancements that do not break existing functionalities. This is a backward-compatible manner when adding functionality.

**Patch:** Used for making backward-compatible bug fixes, appropriate for small changes that fix errors in existing code without adding any new features or changing existing features.

## Installing Changesets:

1.Add to project:
bash

```bash
npm install @changesets/cli --save-dev
# or, if you're using Yarn
yarn add @changesets/cli --dev
```

2. Initialize a changeset:
bash

```bash
npx changeset init
# or
yarn changeset init
```

3. Create a changeset:
bash

```bash
npx changeset
# or
yarn changeset
```

4. Open a Pull Request:

Commit & push changes with the changeset files to your repo, then open a pull request as you normally would. The team can review both code changes & changesets before being merged.

5. Merge and Release:
bash

```bash
npx changeset version
# This will bump package versions and update changelogs according to your changesets

npx changeset publish
# This will publish your packages with the new versions to npm
```

6. Automate with GitHub:
GitHub action to automatically version & publish packages when changes are merged to main.

## Example Release Workflow ymal file

Example of how to extend workflow to include Changeset usage for versioning and optionally, publishing to npm in .github/workflows/ make yml file.

GitHub Action configuration comments on pull requests with changeset summaries and releases changes when Pull Requests are merged:

```yaml
name: Release Workflow

on:
  push:
    branches:
      - stable

jobs:
  release:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@v2

      - name: Setup Node.js
        uses: actions/setup-node@v2
        with:
          node-version: '14'
          registry-url: 'https://registry.npmjs.org/'
      - name: Install dependencies
        run: yarn install

      - name: Create Release Pull Request or Publish to npm
        uses: changesets/action@v1
        with:
          publish: yarn changeset publish
          version: yarn changeset version
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
          NPM_TOKEN: ${{ secrets.NPM_TOKEN }}
```

### Example Separate Versioning and Publishing Workflows

```yaml
# .github/workflows/version.yml
name: Version Workflow
on:
  push:
    branches:
      - main
jobs:
  version:
    runs-on: ubuntu-latest
    steps:
      # Checkout, setup node, install dependencies...
      - name: Bump version and update changelog
        uses: changesets/action@v1
        with:
          version: yarn changeset version
```

```yaml
# .github/workflows/publish.yml
name: Publish Workflow
on:
  push:
    tags:
      - '*'
jobs:
  publish:
    runs-on: ubuntu-latest
    steps:
      # Checkout, setup node, install dependencies...
      - name: Publish to npm
        run: yarn publish
        env:
          NODE_AUTH_TOKEN: ${{ secrets.NPM_TOKEN }}
```