# Contributing Guide

Hi! I’m really excited that you are interested in contributing to ICE. Before submitting your contribution though, please make sure to take a moment and read through the following guidelines.

- [Setup Environment](#setup-environment)

- [Run Examples](#run-examples)

- [Publish Packages](publish-packages)

- [Pull Request Guidelines](#pull-request-guidelines)

- [Issue Reporting Guidelines](#issue-reporting-guidelines)

- [Git Commit Specific](#git-commit-specific)

## Setup Environment

clone repo and initialize the setup environment?

```bash

# 1. clone and setup

$ git clone git@github.com:ice-lab/icejs.git

$ cd icejs && npm run setup

# 2. watch packages

$ npm run watch

```

## Run Examples

We provide a lot of examples, you can run the examples?

```bash

$ cd examples/basic-spa

$ npm link ../../packages/icejs

$ npm start

```

## Publish Packages

When you need to release, you can execute the command?

```bash

$ npm run publish

# 1. ?? ?? ?? Checking the working tree status...

# 2. ? ? ? Building packages...

# 3. ? ? ? Update package version automatically...

# 4. ? ? ? Start publishing...

# 5. ? ? ? Commit & Create tag'...

# 6. ? ? ? Start syncing...

```

\* When you need to release a latest version, the tag will be created automatically, running `npm publish` will tag your package with the `latest` dist-tag.

\* To publish a package with the `beta` dist-tag, you can choose to release rc?beta?alpha versions, the tag will not be created.

## Rollback Packages

When a serious bug occurs in the production environment, you can backtrack the package version?

```bash

# rollback packages

$ npm run rollback <version>

# sync packages

$ npm run sync

```

## Pull Request Guidelines

- Only code that's ready for release should be committed to the master branch. All development should be done in dedicated branches.

- Checkout a \*\*new\*\* topic branch from master branch, and merge back against master branch.

- Make sure `npm test` passes.

- If adding new feature:

- Add accompanying test case.

- Provide convincing reason to add this feature. Ideally you should open a suggestion issue first and have it greenlighted before working on it.

- If fixing a bug:

- If you are resolving a special issue, add `(fix #xxxx[,#xxx])` (#xxxx is the issue id) in your PR title for a better release log, e.g. `update entities encoding/decoding (fix #3899)`.

- Provide detailed description of the bug in the PR. Live demo preferred.

- Add appropriate test coverage if applicable.

## Issue Reporting Guidelines

- The issue list of this repo is \*\*exclusively\*\* for bug reports and feature requests. Non-conforming issues will be closed immediately.

- For simple beginner questions, you can get quick answers from

- For more complicated questions, you can use Google or StackOverflow. Make sure to provide enough information when asking your questions - this makes it easier for others to help you!

- Try to search for your issue, it may have already been answered or even fixed in the development branch.

- It is \*\*required\*\* that you clearly describe the steps necessary to reproduce the issue you are running into. Issues with no clear repro steps will not be triaged. If an issue labeled "need repro" receives no further input from the issue author for more than 5 days, it will be closed.

- For bugs that involves build setups, you can create a reproduction repository with steps in the README.

- If your issue is resolved but still open, don’t hesitate to close it. In case you found a solution by yourself, it could be helpful to explain how you fixed it.

## Git Commit Specific

- Your commits message must follow our [git commit specific](./GIT\_COMMIT\_SPECIFIC.md).

- We will check your commit message, if it does not conform to the specification, the commit will be automatically refused, make sure you have read the specification above.

- You could use `git cz` with a CLI interface to replace `git commit` command, it will help you to build a proper commit-message, see [commitizen](https://github.com/commitizen/cz-cli).

- It's OK to have multiple small commits as you work on your branch - we will let GitHub automatically squash it before merging.