**WindSurf Rules:**  
  
**Two Main Types:**

1. **Global Rules** (global\_rules.md) - Applied across all workspaces
2. **Workspace Rules** (.windsurf/rules) - Applied to specific workspaces/projects

**Rule Storage Locations:**

* .windsurf/rules in your current workspace
* .windsurf/rules in sub-directories
* .windsurf/rules in parent directories (up to git root)
* global\_rules.md for global rules

**Four Activation Modes:**

1. **Manual** - Activated via @mention in Cascade
2. **Always On** - Always applied automatically
3. **Model Decision** - AI decides when to apply based on natural language description
4. **Glob** - Applied to files matching patterns (e.g., \*.js, src/\*\*/\*.ts)

**Access Rules:**

* Click Customizations icon in Cascade's top-right menu
* Navigate to Rules panel
* Create with + Global or + Workspace buttons
* Each rule file limited to 12,000 characters

**Best Practices:**

* Keep rules simple and specific
* Use bullet points and markdown formatting
* Use XML tags to group related rules
* Avoid generic rules (already in AI training)

**Sample rule Format:-**  
  
# General Code Style & Formatting

- Use English for all code and documentation.

- Always declare the type of each variable and function (parameters and return value).

- Create necessary types.

- Use JSDoc to document public classes and methods.

- Don't leave blank lines within a function.

# Naming Conventions

- Use PascalCase for classes.

- Use camelCase for variables, functions, and methods.

- Use kebab-case for file and directory names.

- Use UPPERCASE for environment variables.

- Avoid magic numbers and define constants.

**Getting started with Teams and Enterprise**

Windsurf scales from solo projects to large-scale enterprise codebases. Our Teams and Enterprise plans unlock collaboration features such as team management, Single Sign-On (SSO), advanced analytics, and priority support.

A close-up of a white box

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

For More Info Refer the link <https://docs.windsurf.com/plugins/accounts/teams-getting-started>

**Global Rules Approaches**

**Windsurf offers several approaches for implementing team-wide global rules:  
  
1. Global .codeiumignore Files**

For **enterprise customers**, you can enforce ignore rules across all repositories by placing a global .codeiumignore file in the ~/.codeium/ folder. This applies to all Windsurf workspaces on your system and works alongside repository-specific files.

**2. Global Rules for Cascade**

You can create **global rules** that apply across all workspaces using global\_rules.md files. These rules are manually defined and persist across all conversations and workspaces.

**3. Team Settings Configuration**

**Teams and Enterprise plans** provide centralized configuration through team settings where admins can:

* Configure approved models
* Manage MCP servers
* Set up SSO configurations
* Control role management

**4. User Groups (Enterprise)**

Enterprise plans support **user groups** where you can configure group-specific rules and have group administrators manage rules for their specific teams.

The documentation shows these approaches work at different levels - from individual workspace rules to organization-wide global configurations, with Enterprise plans offering the most comprehensive team-wide rule management capabilities.  
  
  
How to share global rules across the team?

**### Option A: Dedicated Rules Repository (Recommended)**

**\*\*Implementation\*\***: Created complete `team-dev-rules/` structure

- `README.md` - Setup instructions and overview

- `.eslintrc.js` - Centralized ESLint configuration

- `package.json` - Dependencies and scripts

- `setup.js` - Automated installation script

- `global-rules.md` - Team documentation

**\*\*Benefits\*\***:

- One-command setup for new team members

- Centralized rule management

- Automatic enforcement via pre-commit hooks

- Easy updates and versioning

**\*\*Usage\*\***:

```bash

git clone your-repo/team-dev-rules.git

cd your-project

node ../team-dev-rules/setup.js

npm install

```

**### Option B: NPM Package**

**\*\*Use Case\*\***: Multiple projects across organization

```bash

npm publish @your-org/dev-rules

npm install --save-dev @your-org/dev-rules

```

**### Option C: Git Submodule**

**\*\*Use Case\*\***: Related projects in same ecosystem

```bash

git submodule add https://github.com/your-org/team-rules.git .dev-rules

git submodule update --remote

```

**### Option D: Template Repository**

**\*\*Use Case\*\***: Standardized project initialization

- Create GitHub template with pre-configured rules

- Team creates new projects from template