

Types			Commit message structure
<code>feat</code>	Features	A new feature	<p><b>&lt;type&gt;</b>[optional scope]: <b>&lt;description&gt;</b></p> <p>[optional body]</p> <p>[optional footer]</p> <hr/> <p>A commit that has the text <code>BREAKING CHANGE:</code> at the beginning of its commit message introduces a breaking API change</p> <hr/> <p><b>Specification</b></p> <ol style="list-style-type: none"> <li>Commits <b>MUST</b> be prefixed with a type, which consists of a verb, <code>feat</code> and a space.</li> <li>The type <code>feat</code> <b>MUST</b> be used when a commit adds a new feature to the codebase.</li> <li>The type <code>fix</code> <b>MUST</b> be used when a commit represents a bug fix for the codebase.</li> <li>An optional scope <b>MAY</b> be provided after a type. A scope is a phrase describing the part of the codebase enclosed in parenthesis, e.g., <code>fix(parser)</code>.</li> <li>A description <b>MUST</b> immediately follow the type/scope prefix. The description should be a short sentence describing the change, e.g., <code>fix: array parsing issue when multiple items in string</code>.</li> <li>A longer commit body <b>MAY</b> be provided after the short description. The body should be a longer sentence describing the change, e.g., <code>fix: array parsing issue when multiple items in string. This change fixes the issue by parsing the string correctly.</code></li> <li>A footer <b>MAY</b> be provided one blank line after the body. The footer should contain information about the pull-request (such as the issues it fixes, e.g., <code>fix: array parsing issue when multiple items in string. Closes #123</code>).</li> <li>Breaking changes <b>MUST</b> be indicated at the very beginning of the footer. A breaking change <b>MUST</b> consist of the uppercase text <code>BREAKING CHANGE:</code> followed by a space.</li> <li>A description <b>MUST</b> be provided after the <code>BREAKING CHANGE:</code>, describing the API change, e.g., <code>BREAKING CHANGE: environment variables over config files</code>.</li> <li>Types other than <code>feat</code> and <code>fix</code> <b>MAY</b> be used in your commit messages.</li> </ol>
<code>fix</code>	Bug Fixes	A bug fix	
<code>docs</code>	Documentation	Documentation only changes	
<code>style</code>	Styles	Changes that do not affect the meaning of the code (white-space, formatting, missing semi-colons, etc)	
<code>refactor</code>	Code Refactoring	A code change that neither fixes a bug nor adds a feature	
<code>perf</code>	Performance Improvements	A code change that improves performance	
<code>test</code>	Tests	Adding missing tests or correcting existing tests	
<code>build</code>	Builds	Changes that affect the build system or external dependencies (example scopes: gulp, broccoli, npm)	
<code>ci</code>	Continuous Integrations	Changes to our CI configuration files and scripts (example scopes: Travis, Circle, BrowserStack, SauceLabs)	
<code>chore</code>	Chores	Other changes that don't modify src or test files	
<code>revert</code>	Reverts	Reverts a previous commit	

