

9. Using Vue in Markdown by Vicente v2

default theme

- 9. Using Vue in Markdown by Vicente v2 default theme
 - new! 9.1. Browser API Access Restrictions
 - 9.2. Browser API Access Restrictions
 - 9.3. Browser API Access Restrictions
 - 9.4. Browser API Access Restrictions
 - 9.5. Browser API Access Restrictions
 - 9.6. Browser API Access Restrictions
 - 9.7. Browser API Access Restrictions

new! 9.1. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#) . In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

<code>java.lang.ProcessBuilder</code>	<code>java.lang.Process</code>
<div>+ directory: File</div> <div>+ redirectErrorStream: boolean</div> <div>+ ProcessBuilder(command: String[])</div> <div>+ ProcessBuilder(command: List<String>)</div> <div>+ command(): List</div> <div>+ command(command: List<String>): ProcessBuilder</div> <div>+ command(command: String[]): ProcessBuilder</div> <div>+ directory(): File</div> <div>+ directory(directory: File): ProcessBuilder</div> <div>+ environment(): Map</div> <div>+ redirectErrorStream(): boolean</div> <div>+ redirectErrorStream(redirectErrorStream: boolean): ProcessBuilder</div> <div>+ start(): Process</div>	<div>+ destroy()</div> <div>+ exitValue(): int</div> <div>+ getErrorStream(): InputStream</div> <div>+ getInputStream(): InputStream</div> <div>+ getOutputStream(): OutputStream</div> <div>+ waitFor(): int</div>

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component:

PHP Example	Java Title	
1	# Post Here Code Example	php
1	// Post Here Code Example	java

9.2. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#) . In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component:

PRUEBA

This is a tip

9.3. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#) . In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component:

9.4. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#) . In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component:

9.5. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#) . In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component:

9.6. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#). In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component:

9.7. Browser API Access Restrictions

Because VuePress applications are server-rendered in Node.js when generating static builds, any Vue usage must conform to the [universal code requirements](#). In short, make sure to only access Browser / DOM APIs in `beforeMount` or `mounted` hooks.

If you are using or demoing components that are not SSR friendly (for example containing custom directives), you can wrap them inside the built-in `<ClientOnly>` component: