SVELTE • RUNTIME

Imperative component API

ON THIS PAGE

Every Svelte application starts by imperatively creating a root component. On the client this component is mounted to a specific element. On the server, you want to get back a string of HTML instead which you can render. The following functions help you achieve those tasks.

mount

Instantiates a component and mounts it to the given target:

```
import { mount } from 'svelte';
import App from './App.svelte';

const app = mount(App, {
  target: document.querySelector('#app'),
  props: { some: 'property' }
});
```

You can mount multiple components per page, and you can also mount from within your application, for example when creating a tooltip component and attaching it to the hovered element.

Note that unlike calling `new App(...)` in Svelte 4, things like effects (including `onMount` callbacks, and action functions) will not run during `mount`. If you need to force pending effects to run (in the context of a test, for example) you can do so with `flushSync()`.

Unmounts a component created with `mount` or `hydrate`:

```
import { mount, unmount } from 'svelte';
import App from './App.svelte';

const app = mount(App, {...});

// later
unmount(app);
```

render

Only available on the server and when compiling with the `server` option. Takes a component and returns an object with `body` and `head` properties on it, which you can use to populate the HTML when server-rendering your app:

```
import { render } from 'svelte/server';
import App from './App.svelte';

const result = render(App, {
  props: { some: 'property' }
});
result.body; // HTML for somewhere in this <body> tag
result.head; // HTML for somewhere in this <head> tag
```

hydrate

Like `mount`, but will reuse up any HTML rendered by Svelte's SSR output (from the `render` function) inside the target and make it interactive:

```
import { hydrate } from 'svelte';
import App from './App.svelte';
```

As with `mount`, effects will not run during `hydrate` — use `flushSync()` immediately afterwards if you need them to.

Edit this page on GitHub

PREVIOUS
Lifecycle hooks

NEXT
Testing