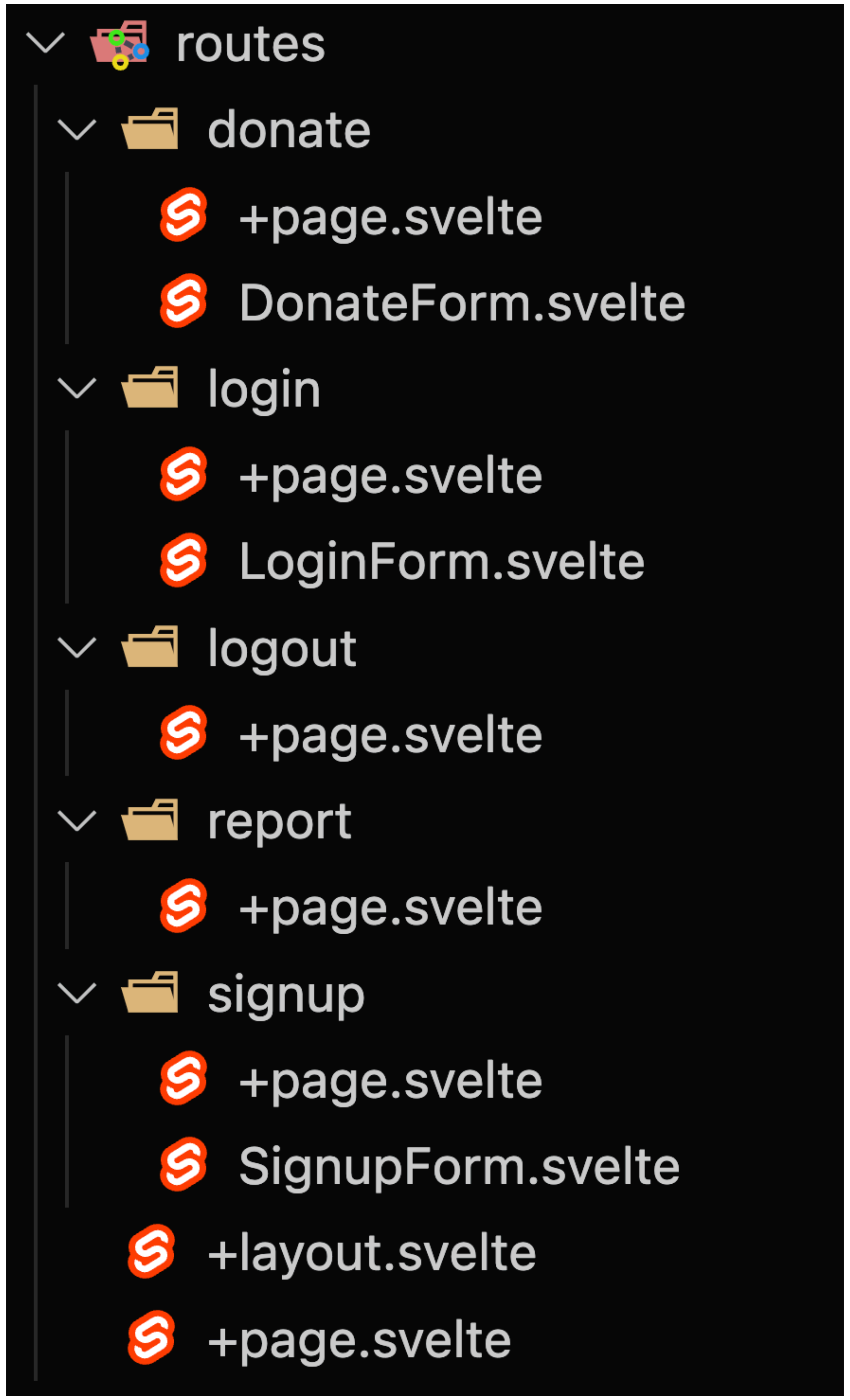
PageLoad in SSR

PageLoad & PageServer Load functions

SvelteKit Page Components

- At the heart of SvelteKit is a filesystem-based router.
- The routes of your app — i.e. the URL paths that users can access — are defined by the directories in your codebase:
  - src/routes is the root route
  - src/routes/report creates a /report route

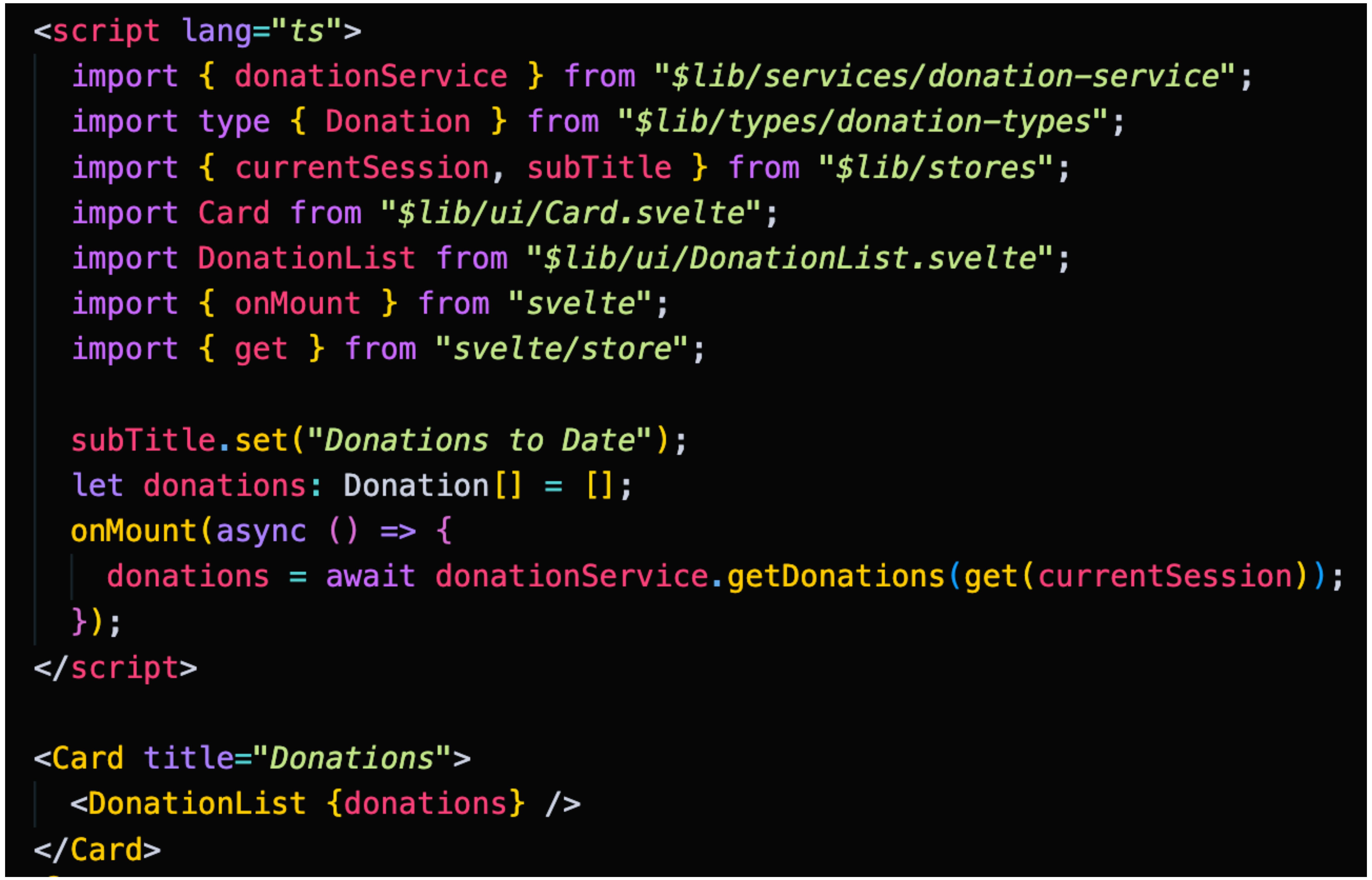
Each route directory contains one or more route files, which can be identified by their + prefix.



```
routes
  donate
    +page.svelte
    DonateForm.svelte
  login
    +page.svelte
    LoginForm.svelte
  logout
    +page.svelte
  report
    +page.svelte
  signup
    +page.svelte
    SignupForm.svelte
  +layout.svelte
  +page.svelte
```

+page.svelte

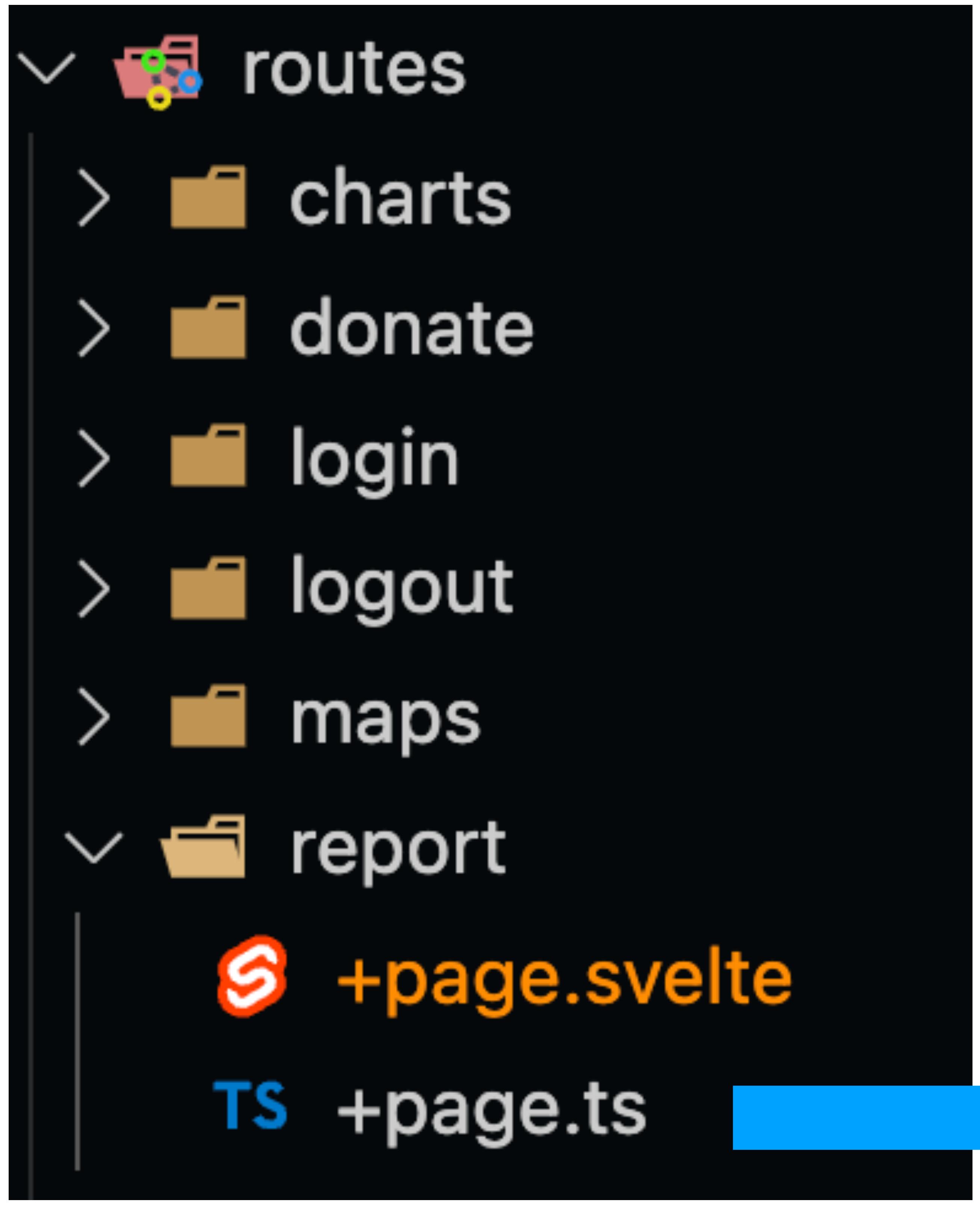
- A +page.svelte component defines a page of your app.



```
<script lang="ts">
  import { donationService } from "$lib/services/donation-service";
  import type { Donation } from "$lib/types/donation-types";
  import { currentSession, subTitle } from "$lib/stores";
  import Card from "$lib/ui/Card.svelte";
  import DonationList from "$lib/ui/DonationList.svelte";
  import { onMount } from "svelte";
  import { get } from "svelte/store";

  subTitle.set("Donations to Date");
  let donations: Donation[] = [];
  onMount(async () => {
    donations = await donationService.getDonations(get(currentSession));
  });
</script>

<Card title="Donations">
  <DonationList {donations} />
</Card>
```

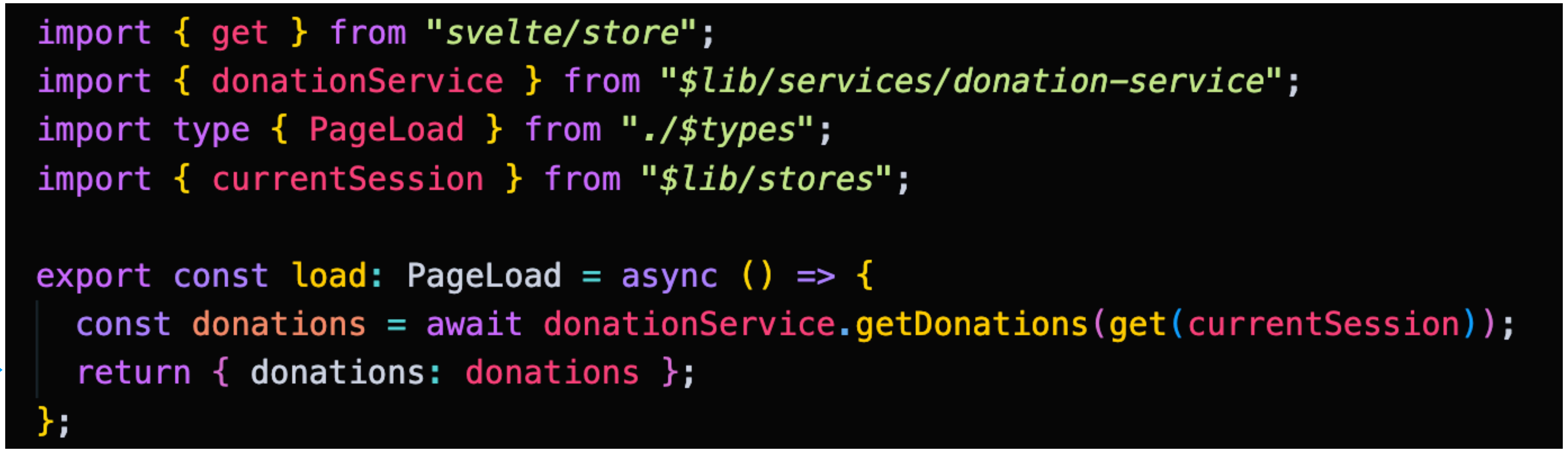
By default, pages are rendered both on the server (SSR) for the initial request and in the browser (CSR) for subsequent navigation

+page.ts



- Often, a page will need to load some data before it can be rendered.

- For this, we add a +page.ts module that exports a load function:

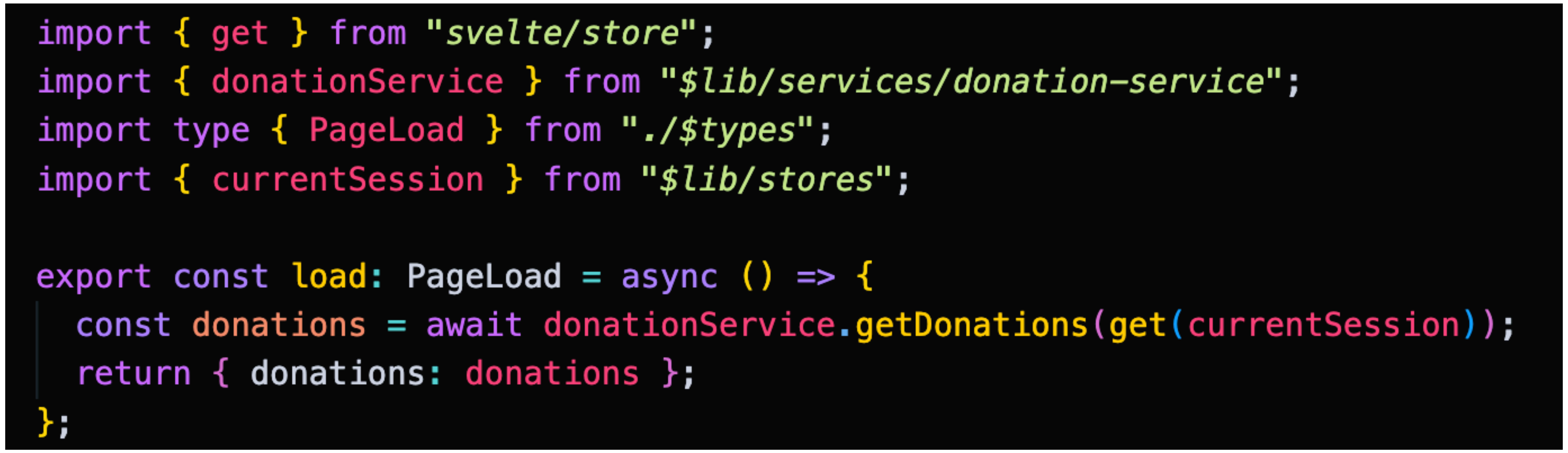


```
import { get } from "svelte/store";
import { donationService } from "$lib/services/donation-service";
import type { PageLoad } from "./$types";
import { currentSession } from "$lib/stores";

export const load: PageLoad = async () => {
  const donations = await donationService.getDonations(get(currentSession));
  return { donations: donations };
};
```

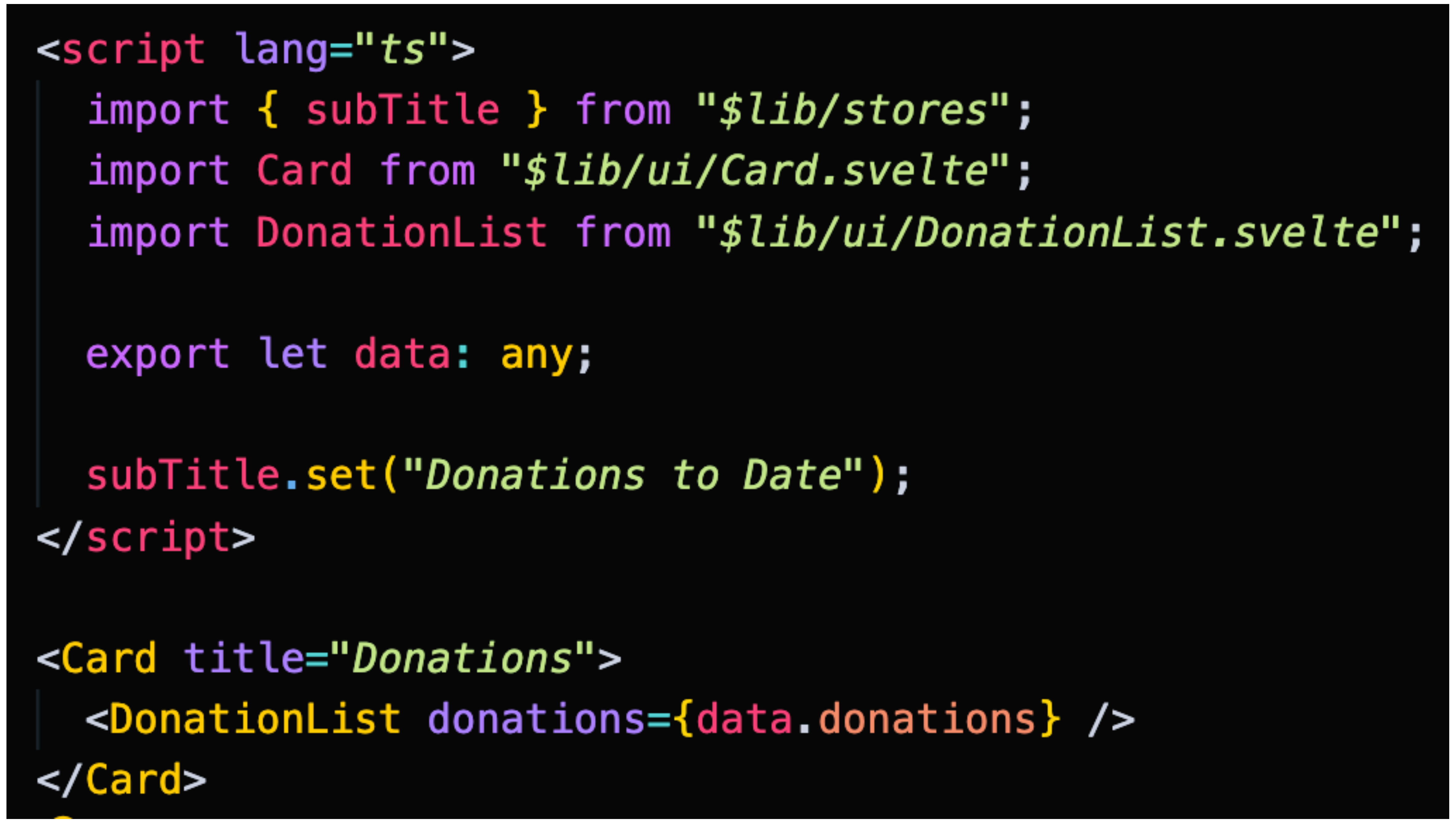
This function runs alongside +page.svelte, which means it runs on the server during server-side rendering and in the browser during client-side navigation

- **load** function reads donations from the donationService (API)

+page.ts



```
import { get } from "svelte/store";
import { donationService } from "$lib/services/donation-service";
import type { PageLoad } from "./$types";
import { currentSession } from "$lib/stores";

export const load: PageLoad = async () => {
  const donations = await donationService.getDonations(get(currentSession));
  return { donations: donations };
};
```



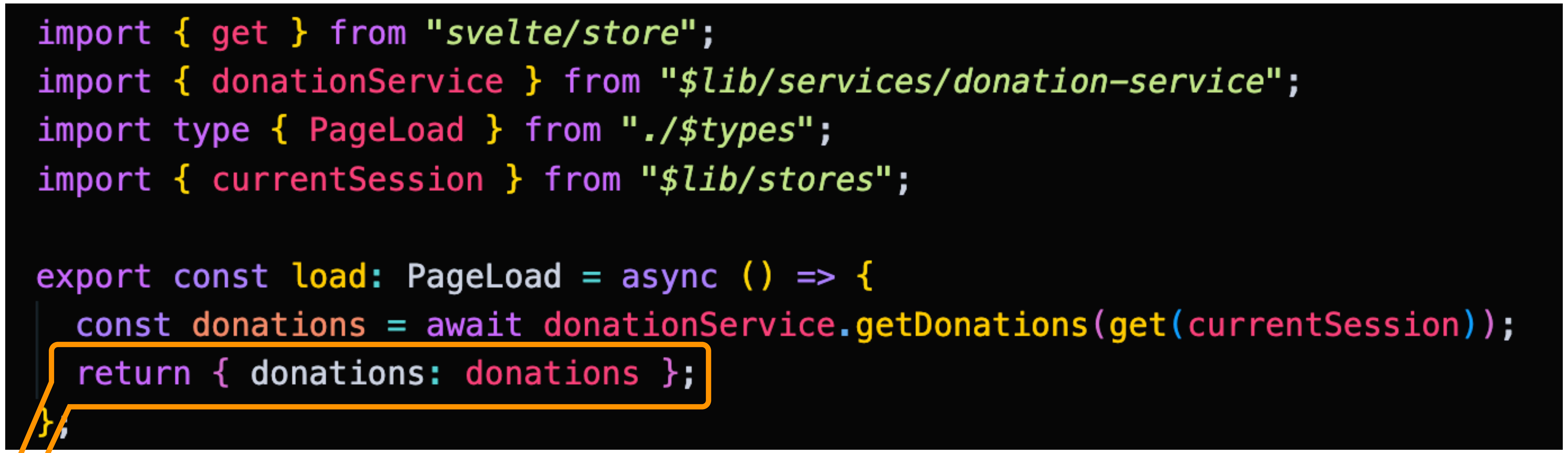
```
<script lang="ts">
  import { subTitle } from "$lib/stores";
  import Card from "$lib/ui/Card.svelte";
  import DonationList from "$lib/ui/DonationList.svelte";

  export let data: any;

  subTitle.set("Donations to Date");
</script>

<Card title="Donations">
  <DonationList donations={data.donations} />
</Card>
```

+page.svelte

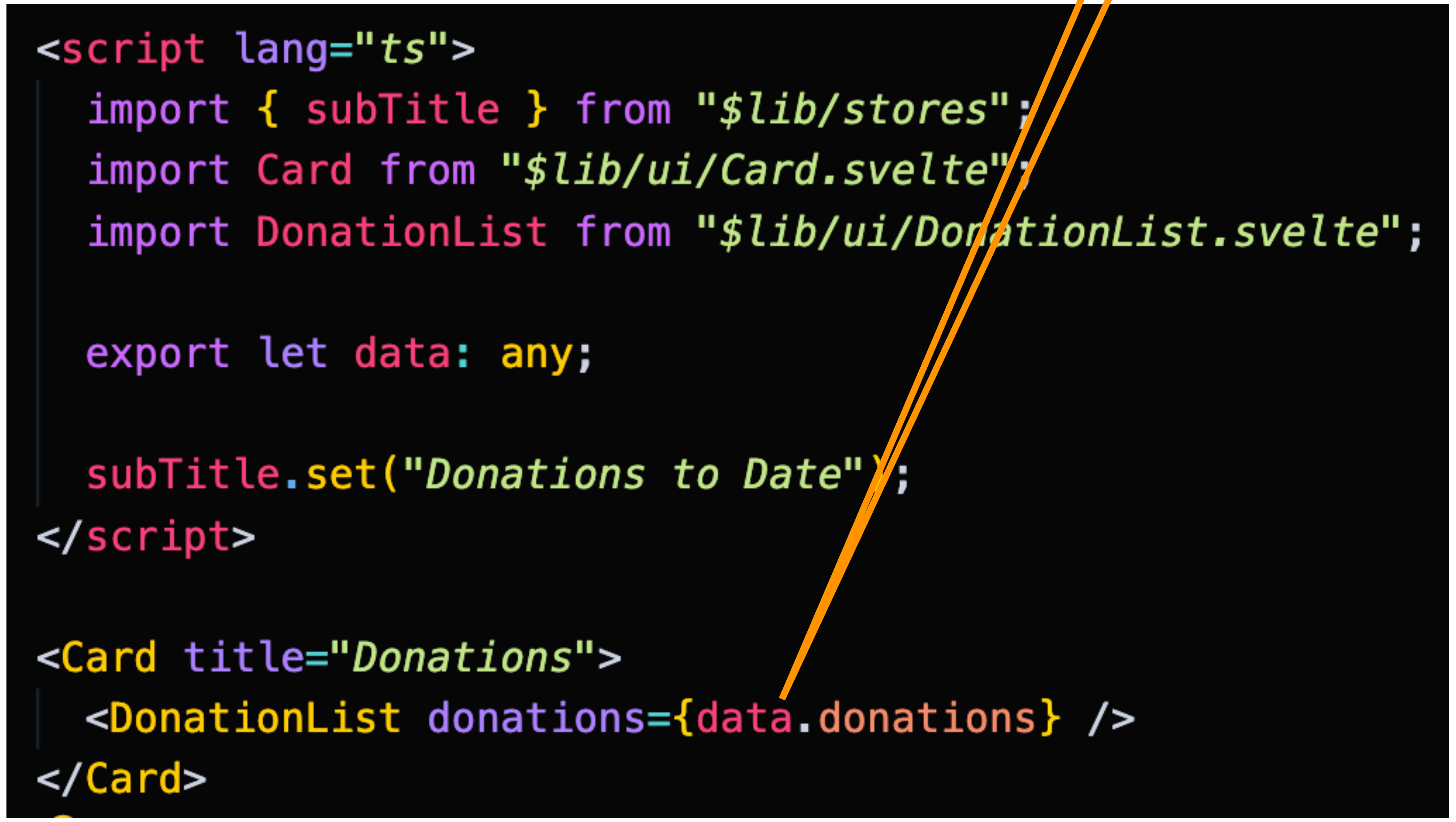
- **data** object contains values returned from the **load** function

+page.ts

- **load** function reads donations from the donationService (API)



```ts
import { get } from "svelte/store";
import { donationService } from "$lib/services/donation-service";
import type { PageLoad } from "./$types";
import { currentSession } from "$lib/stores";

export const load: PageLoad = async () => {
  const donations = await donationService.getDonations(get(currentSession));
  return { donations: donations };
};
```



```svelte
<script lang="ts">
  import { subTitle } from "$lib/stores";
  import Card from "$lib/ui/Card.svelte";
  import DonationList from "$lib/ui/DonationList.svelte";

  export let data: any;

  subTitle.set("Donations to Date");
</script>

<Card title="Donations">
  <DonationList donations={data.donations} />
</Card>
```

+page.svelte

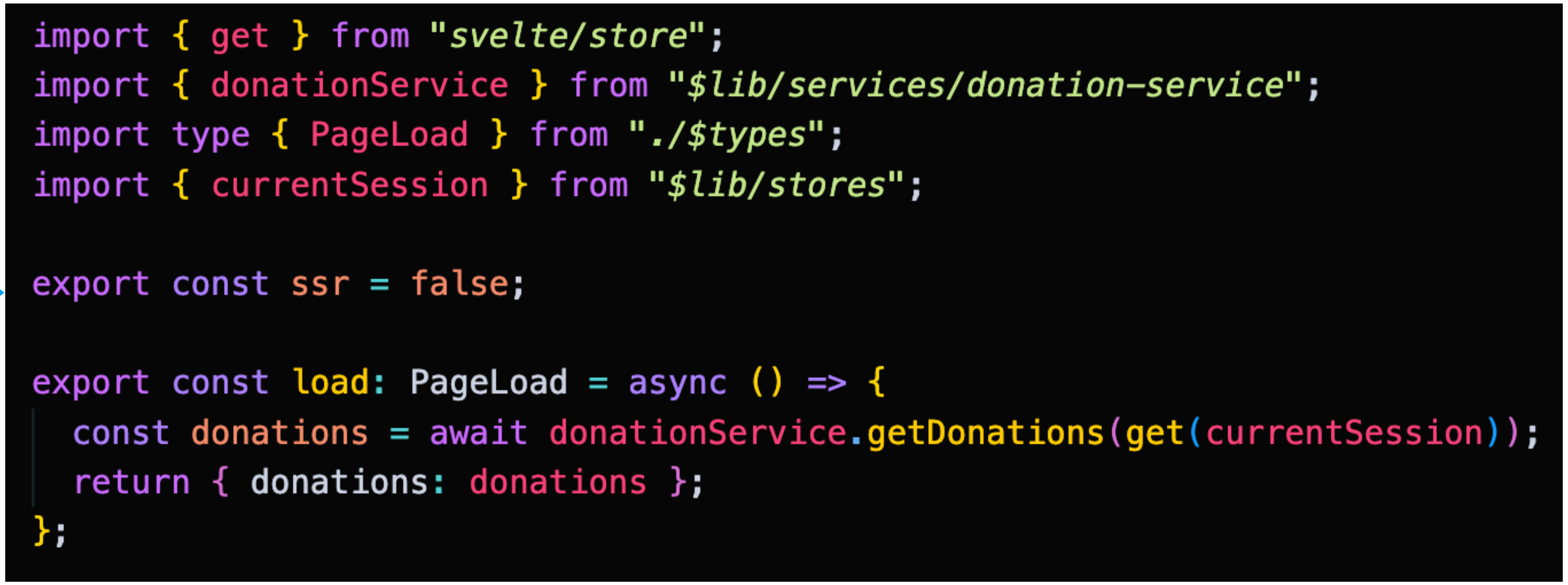
- **data** object contains values returned from the **load** function

Customise Rendering Behaviour

- As well as **load,** +page.ts can export values that configure the page's behaviour.

- SSR set to false forces all rendering to take place in the browser.

- i.e. page.ts will always run win the browser



```
import { get } from "svelte/store";
import { donationService } from "$lib/services/donation-service";
import type { PageLoad } from "./$types";
import { currentSession } from "$lib/stores";

export const ssr = false;

export const load: PageLoad = async () => {
  const donations = await donationService.getDonations(get(currentSession));
  return { donations: donations };
};
```

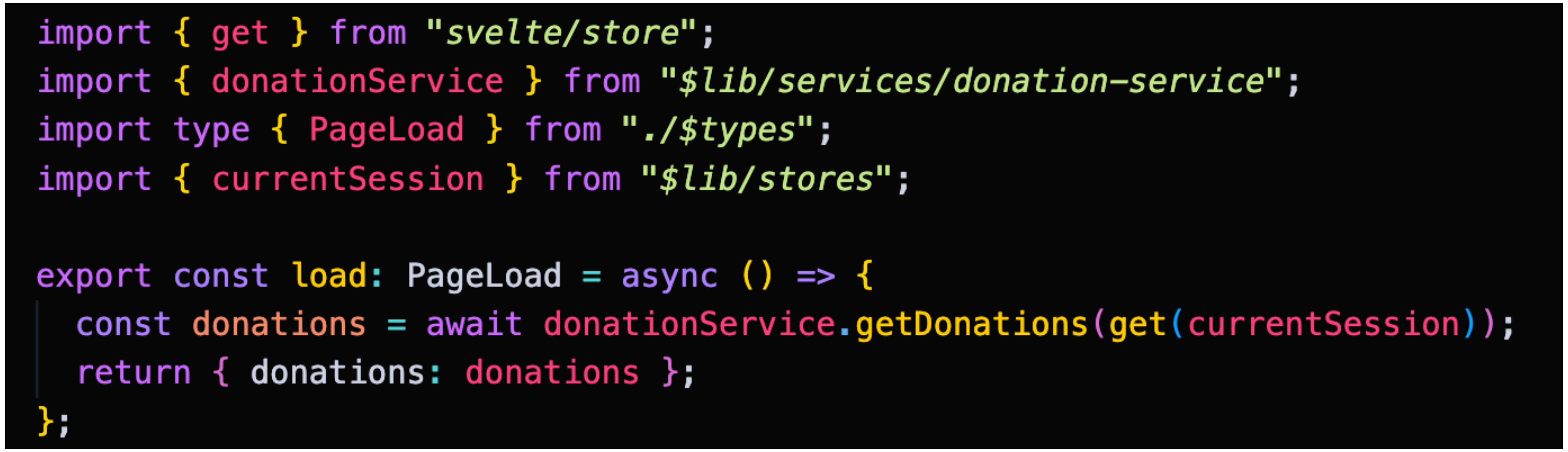
export const ssr = false;

- If the load function can only run on the server rename:

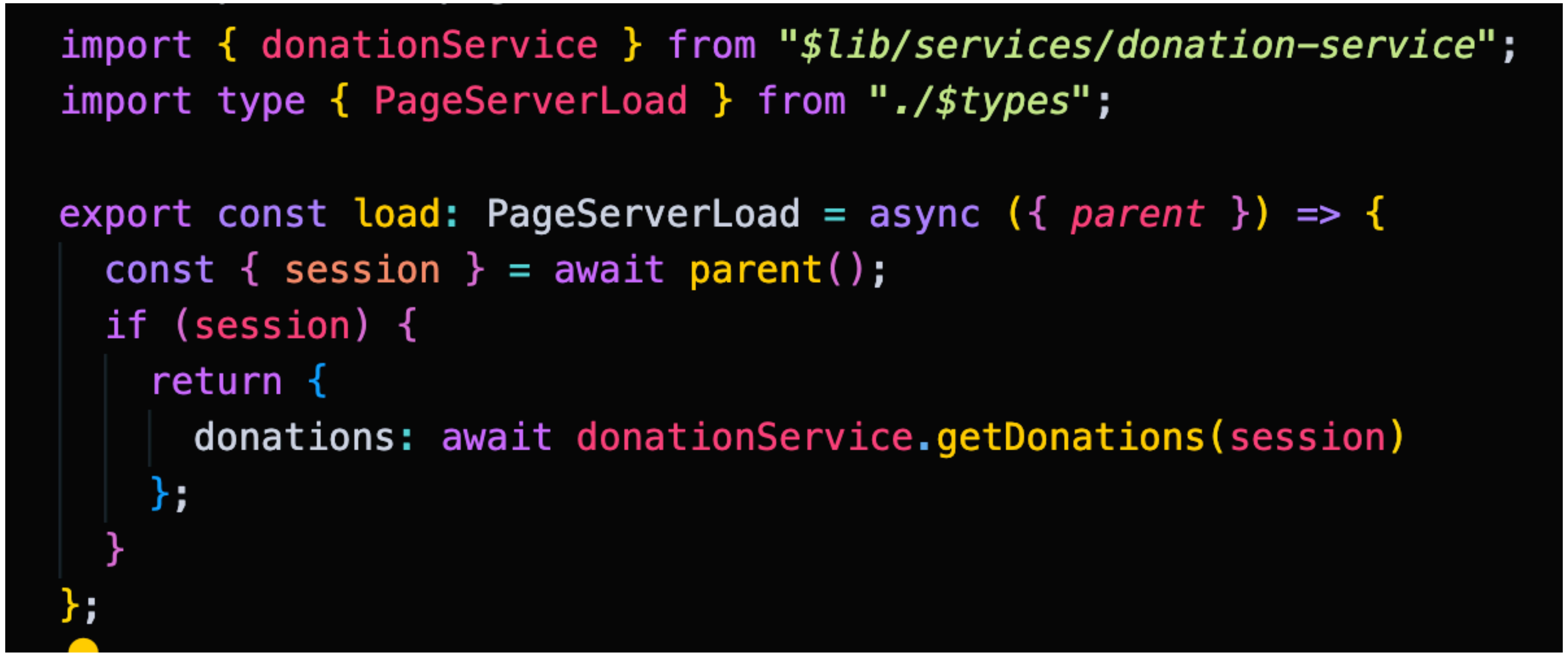
  - +page.ts to +**page.server.ts**

  - PageLoad type to PageServerLoad.

+page.ts



```
import { get } from "svelte/store";
import { donationService } from "$lib/services/donation-service";
import type { PageLoad } from "./$types";
import { currentSession } from "$lib/stores";

export const load: PageLoad = async () => {
  const donations = await donationService.getDonations(get(currentSession));
  return { donations: donations };
};
```



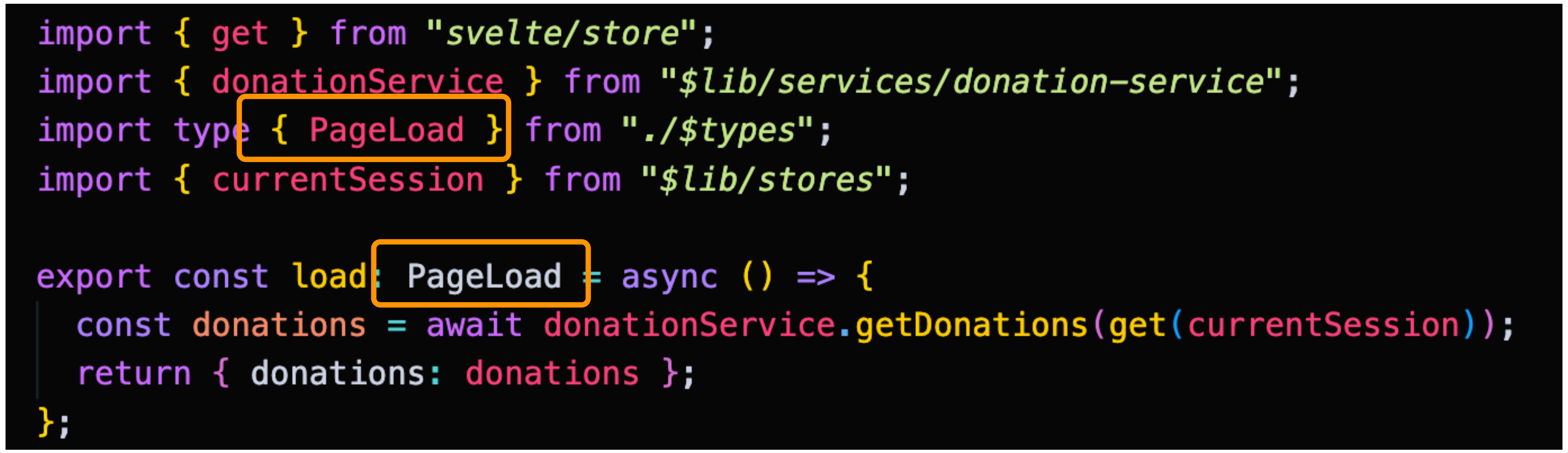
```
import { donationService } from "$lib/services/donation-service";
import type { PageServerLoad } from "./$types";

export const load: PageServerLoad = async ({ parent }) => {
  const { session } = await parent();
  if (session) {
    return {
      donations: await donationService.getDonations(session)
    };
  }
};
```

+page.server.ts

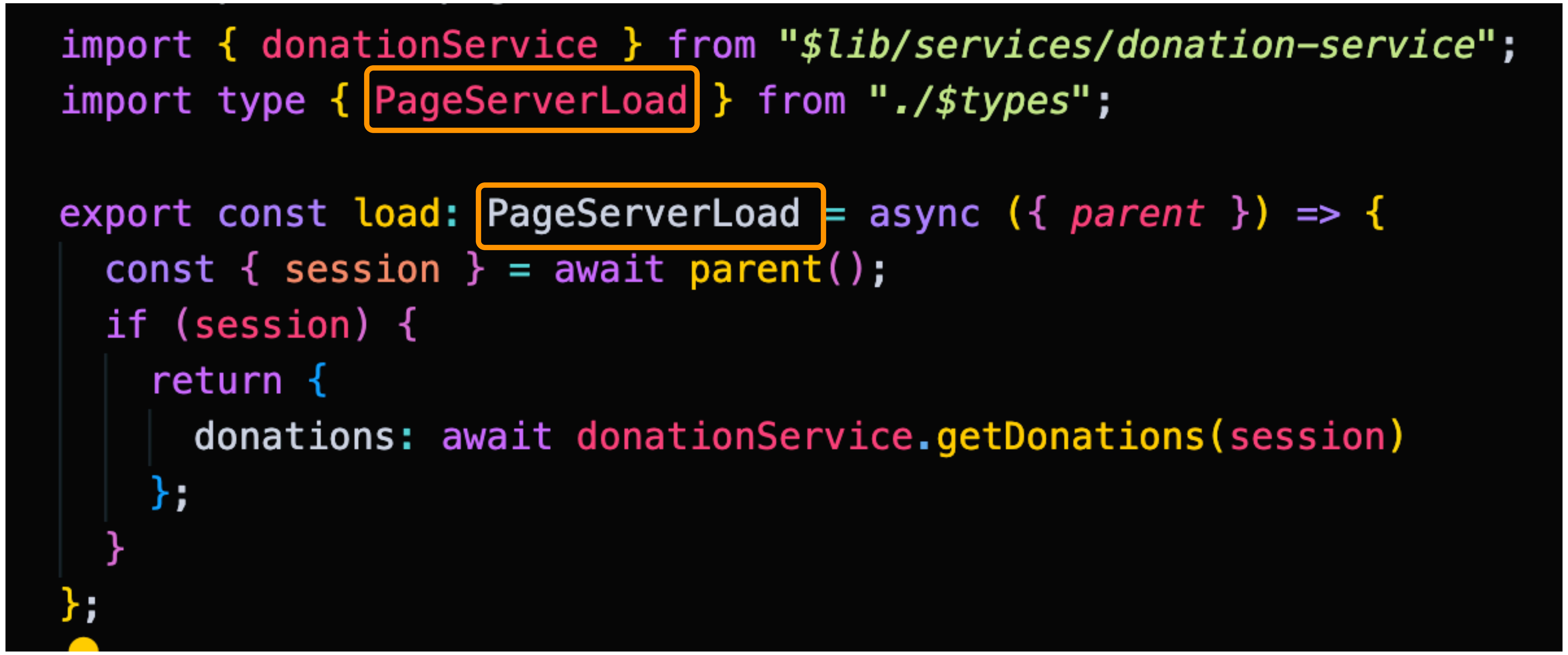
+page.ts

- This may be appropriate if the app needs to fetch data from a database

- Or you need to access private environment variables like API keys



```
import { get } from "svelte/store";
import { donationService } from "$lib/services/donation-service";
import type { PageLoad } from "./$types";
import { currentSession } from "$lib/stores";

export const load: PageLoad = async () => {
  const donations = await donationService.getDonations(get(currentSession));
  return { donations: donations };
};
```



```
import { donationService } from "$lib/services/donation-service";
import type { PageServerLoad } from "./$types";

export const load: PageServerLoad = async ({ parent }) => {
  const { session } = await parent();
  if (session) {
    return {
      donations: await donationService.getDonations(session)
    };
  }
};
```

+page.server.ts

PageLoad in SSR

PageLoad & PageServer Load functions