Alaa Asfahani

Angular

VERSION 17

New features and updates

WHAT’S NEW?

Angular 17 introduces a refreshed brand and a new documentation website, making it easier for developers to learn and use Angular.

Angular 17 introduces a new control flow syntax @if, @for and @switch that simplifies template iteration, making it more natural and faster.

Angular 17's "defer views" feature allows for lazy loading of components, improving page loading times and user experience.

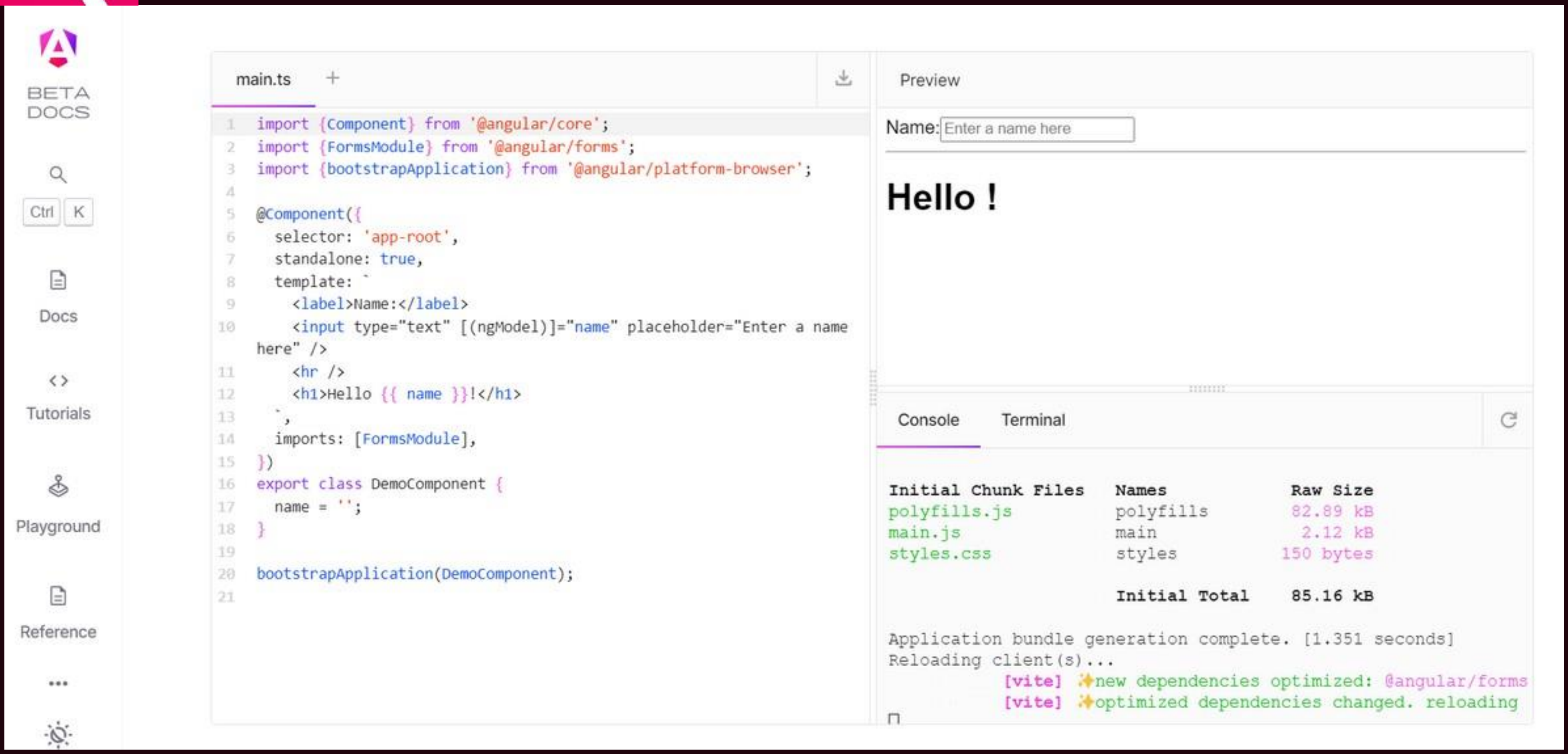
Hydration in Angular is now stable for server-side rendering, and Hybrid Rendering with SSG is enhanced in v17

Standalone apps by default, esbuild, and Vite integration lead to faster build times and performance improvements.

NEW DOCUMENTATION

Angular 17 offers an interactive tutorial, video tutorial, and standard documentation to cater to different learning preferences. All information there is revised and updated.

Now Angular.dev is the new home for Angular development. It provides an interactive web container on the website where developers can try Angular and a step-by-step interactive tutorial to learn the framework.



```
import {Component} from '@angular/core';
import {FormsModule} from '@angular/forms';
import {bootstrapApplication} from '@angular/platform-browser';

@Component({
  selector: 'app-root',
  standalone: true,
  template: `
    <label>Name:</label>
    <input type="text" [(ngModel)]="name" placeholder="Enter a name
here" />
    <hr />
    <h1>Hello {{ name }}!</h1>
  `,
  imports: [FormsModule],
})
export class DemoComponent {
  name = '';
}

bootstrapApplication(DemoComponent);
```

Name: Enter a name here

Hello !

```
Initial Chunk Files   Names            Raw Size
polyfills.js          polyfills        82.89 kB
main.js               main              2.12 kB
styles.css            styles          150 bytes

                      Initial Total    85.16 kB

Application bundle generation complete. [1.351 seconds]
Reloading client(s)...
           [vite] ✨new dependencies optimized: @angular/forms
           [vite] ✨optimized dependencies changed. reloading
```

NEW CONTROL SYSTEM

Angular v17 introduces a new control flow syntax that significantly enhances the way developers can write if statements, for loops, and switch statements within their templates. This new syntax not only simplifies the process but also makes reading and writing logic and templates more intuitive.

The new control flow syntax in Angular uses template keywords starting with an @ symbol, making the code clearer and more robust compared to NgIf-based syntax.

@if @switch

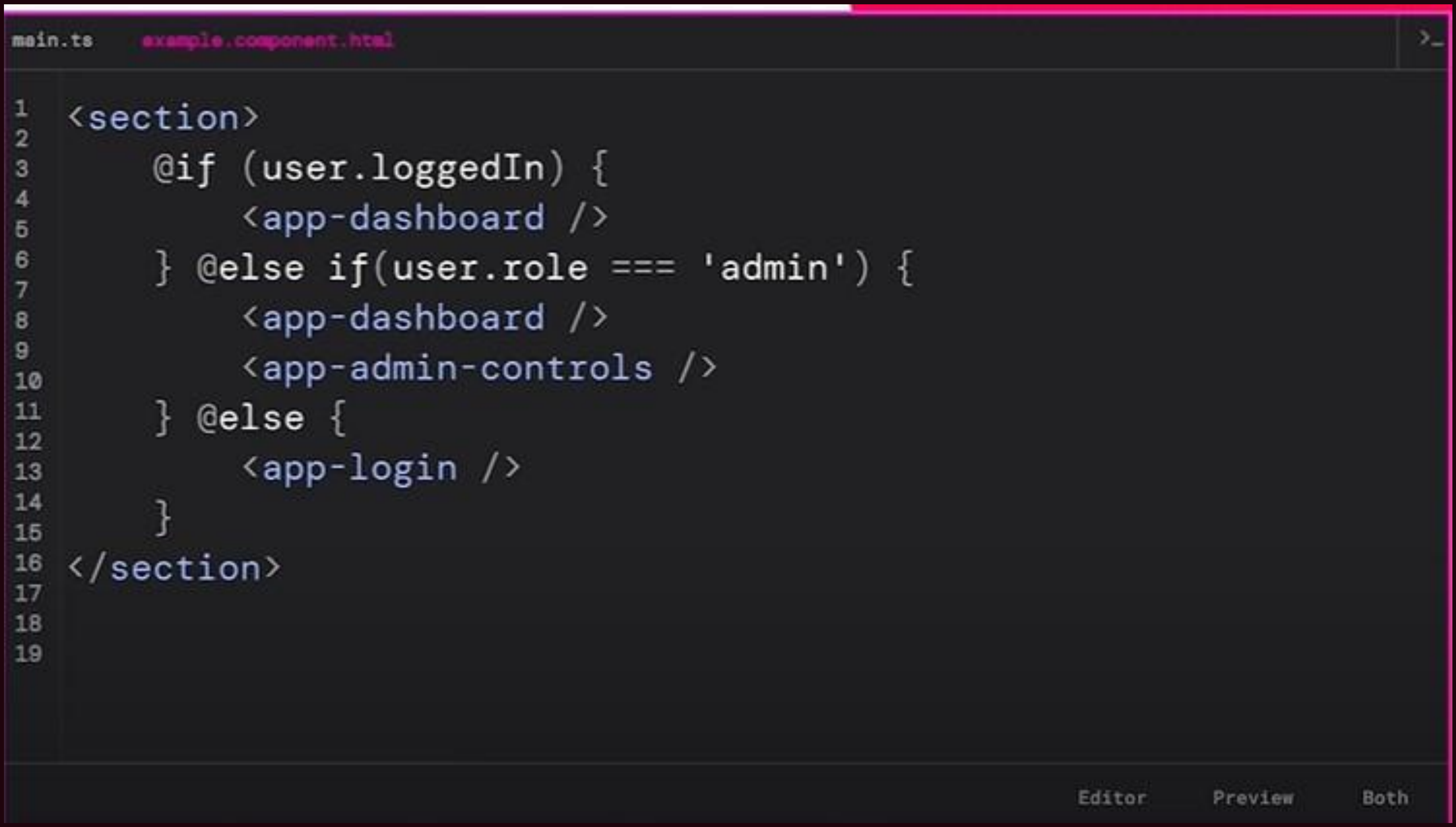
@for

@IF

Traditionally, Angular used NgIf for conditional rendering, but it had its limitations, particularly in handling complex conditional logic and rendering multiple elements. The new "@if" syntax addresses these issues head-on.

This new approach mirrors the simplicity and intuitiveness of JavaScript's if statement, making it instantly familiar to developers.

What makes this update more exciting is the introduction of "else if" and "else" support, which was a much-requested feature. Now, developers can handle multiple conditions seamlessly without resorting to additional Ng templates.



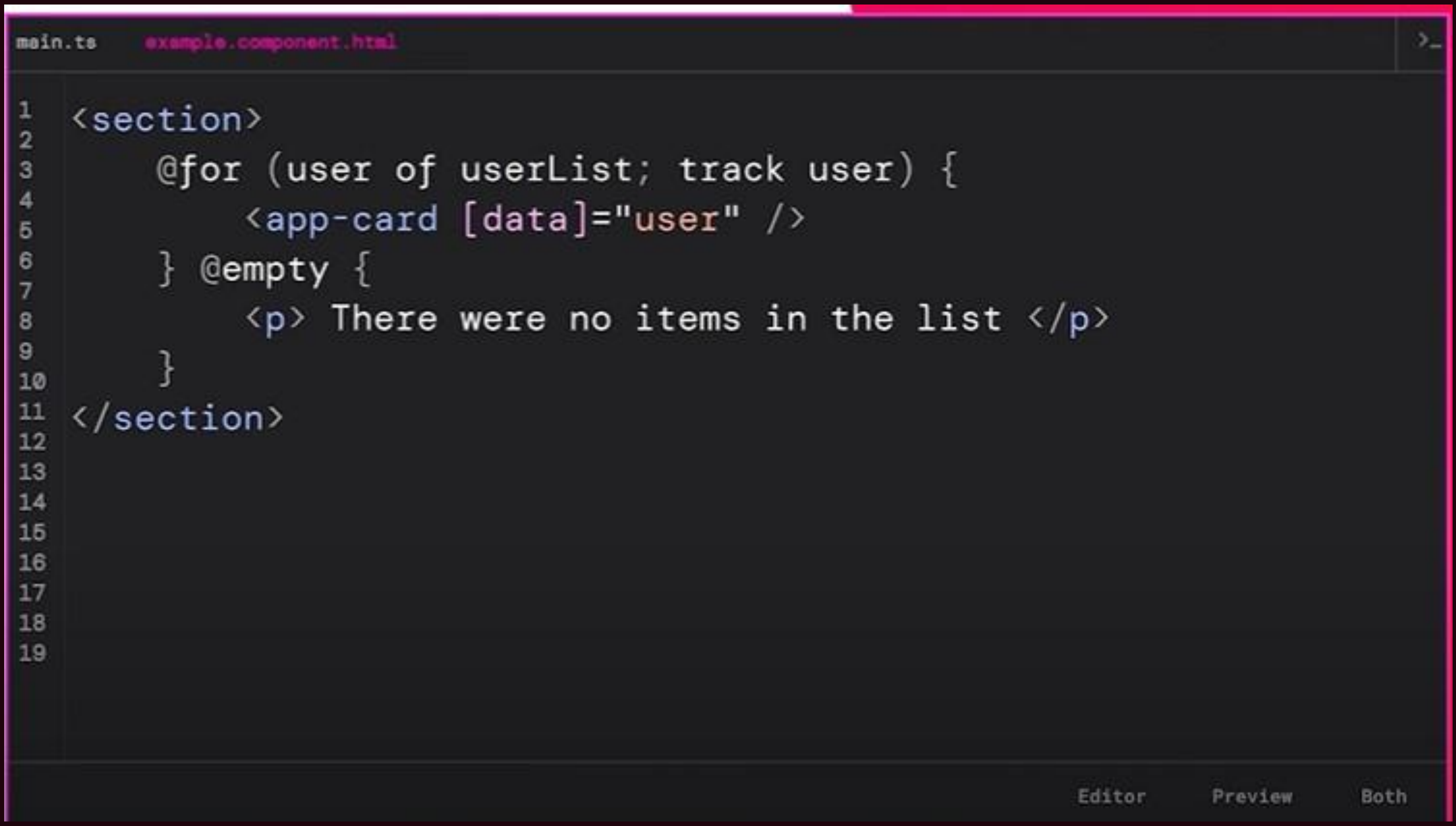
```
<section>
    @if (user.loggedIn) {
        <app-dashboard />
    } @else if(user.role === 'admin') {
        <app-dashboard />
        <app-admin-controls />
    } @else {
        <app-login />
    }
</section>
```

@FOR

Looping in Angular templates has been updated too with   Angular v17's **@for** syntax.

In previous Angular versions, developers often faced challenges with loop efficiency, especially when dealing with dynamic lists. The **@for** syntax addresses this by allowing for more optimized rendering of lists.

This change significantly speeds up loop execution and makes handling list updates more efficient. Additionally, Angular v17 introduces the **@empty** syntax, a convenient way to handle scenarios where lists are empty. This eliminates the need for complex conditional templates and further streamlines development.

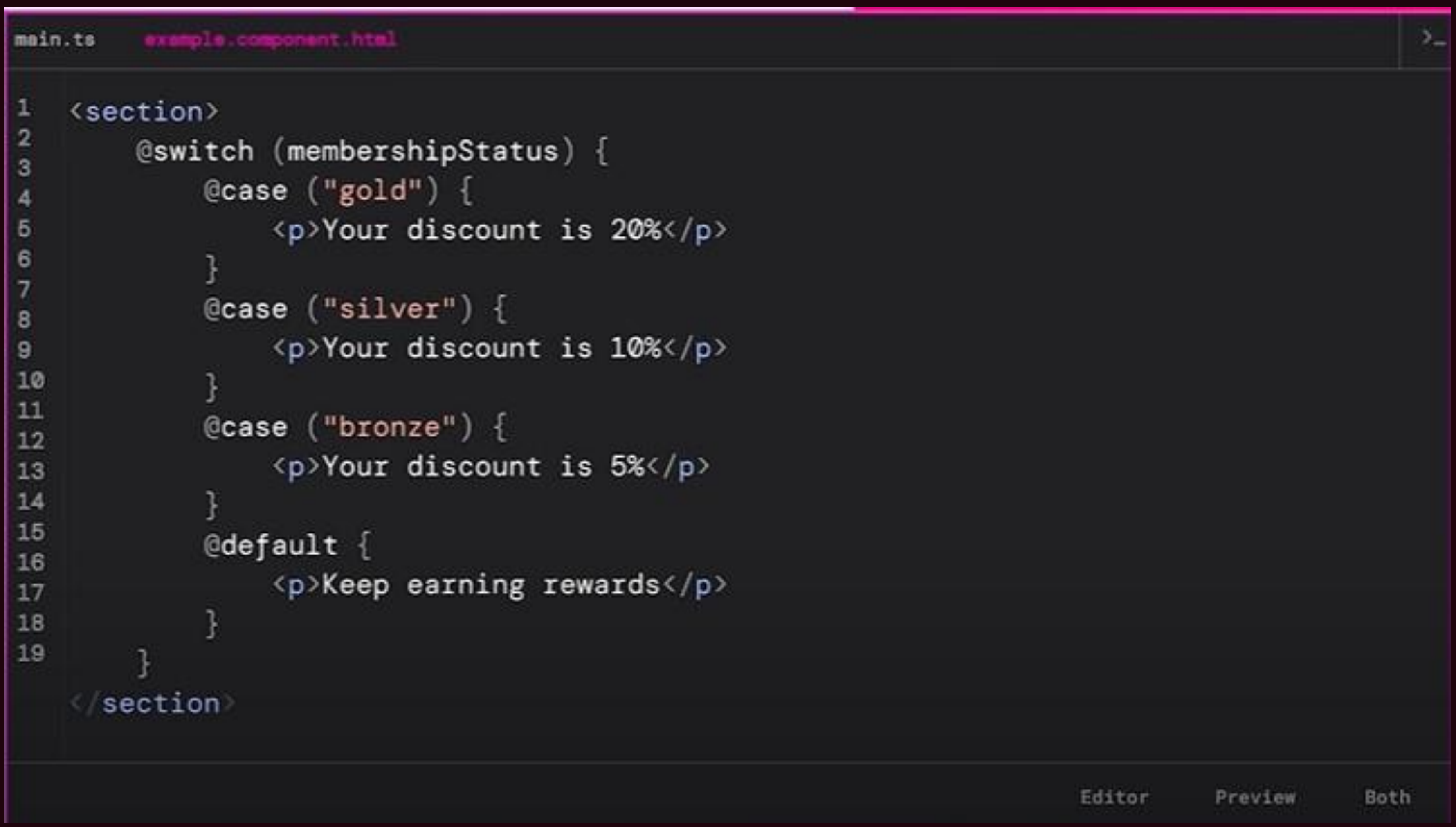


```
<section>
    @for (user of userList; track user) {
        <app-card [data]="user" />
    } @empty {
        <p> There were no items in the list </p>
    }
</section>
```

@SWITCH

Switch statements in Angular templates have been made more robust and intuitive with the **@switch** syntax in Angular v17. Previously, implementing switch-case logic in templates could be cumbersome and less readable. The new **@switch** syntax directly addresses these issues by closely mirroring JavaScript’s switch statement structure.

This syntax uses **@case** and **@default** keywords, allowing developers to clearly define various cases within their templates.



```
<section>
    @switch (membershipStatus) {
        @case ("gold") {
            <p>Your discount is 20%</p>
        }
        @case ("silver") {
            <p>Your discount is 10%</p>
        }
        @case ("bronze") {
            <p>Your discount is 5%</p>
        }
        @default {
            <p>Keep earning rewards</p>
        }
    }
</section>
```

This syntax is not only more straightforward but also improves the readability and maintainability of the code. Developers can now implement switch-case logic directly within their templates, making the code cleaner and more concise.

DEFERRED LOADING

The new "defer block" feature, now available in developer preview, simplifies this process. It allows developers to designate specific chunks of content within a component's template for deferred loading. This feature is particularly useful for large components that are not needed as part of the initial bundle.

To implement this, Angular introduces a block syntax for templates. Within this block, developers can use various triggers to specify when deferred loading should occur.

These triggers include: **defer on viewport:** Triggers loading when an element enters the viewport.

defer on idle: Loads content when the browser is idle. defer on interaction: Loads upon user interactions like clicks or focus.

defer on hover: Triggers when the mouse hovers over a specified area. **defer on timer:** Loads after a set timeout period.

defer on immediate: Immediately loads after rendering the defer block.

Furthermore, developers have the flexibility to create custom triggers with the "when" clause, ensuring that deferred loading perfectly aligns with the specific needs of their applications.

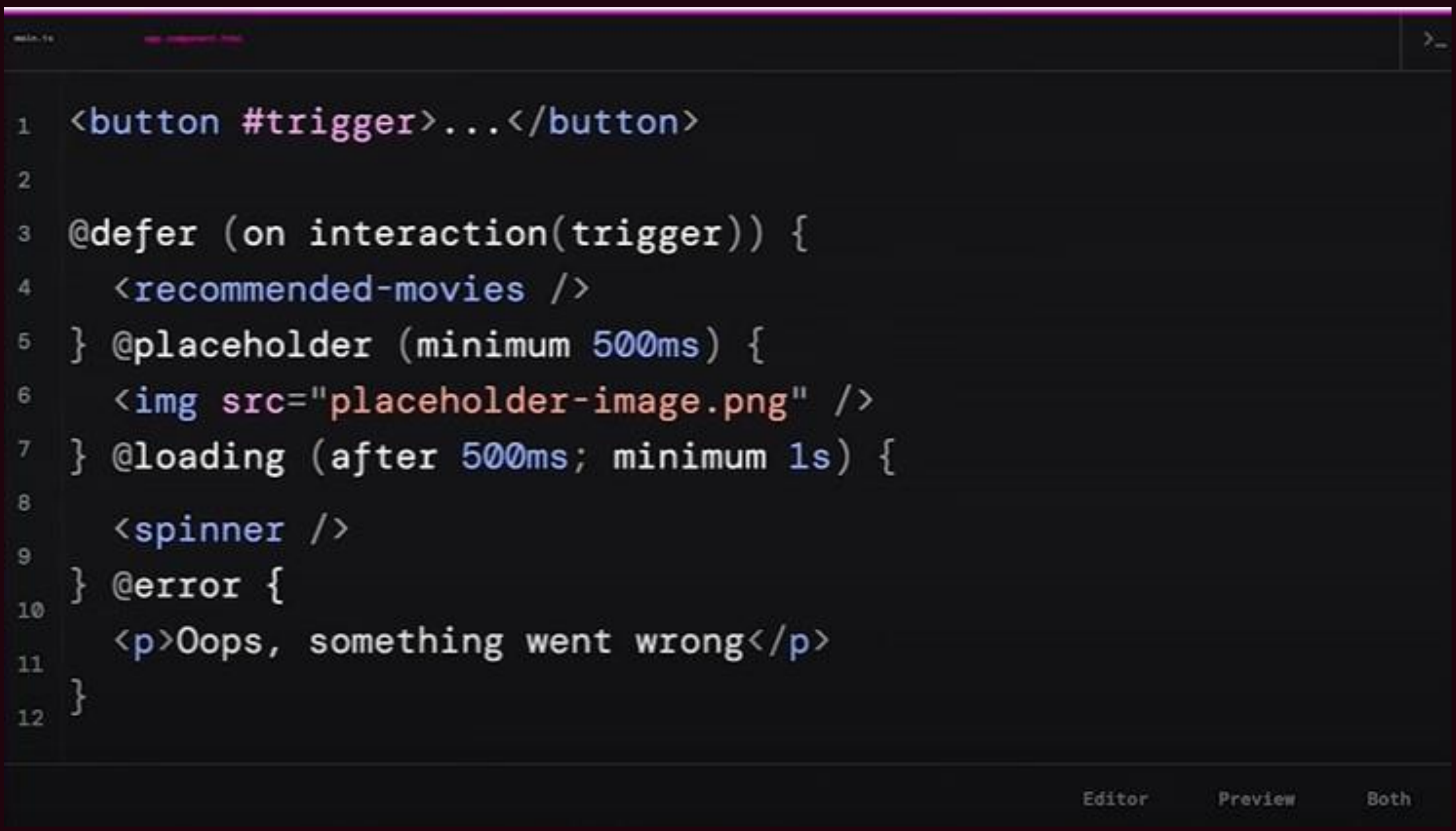
@DEFER

Angular v17's deferred loading extends beyond basic functionality, offering developers enhanced control over the loading process. This includes the ability to prefetch dependencies using ‘prefetch’ on or ‘prefetch when’, preparing resources in advance based on specified conditions or custom criteria.
The defer blocks also provide sections for various loading phases.

For instance:

@placeholder: Shows a placeholder before the deferred content loads.
@loading: Displays a loading spinner during content fetching, with options to set minimum and maximum display times to enhance user experience.
@error: A designated area for displaying content if an error occurs during loading.

4



```
<button #trigger>...</button>

@defer (on interaction(trigger)) {
  <recommended-movies />
} @placeholder (minimum 500ms) {
  <img src="placeholder-image.png" />
} @loading (after 500ms; minimum 1s) {
  <spinner />
} @error {
  <p>Oops, something went wrong</p>
}
```

HYDRATION FEATURE

Hydration has now achieved stability and is production-ready in Angular version 17. This marks a significant milestone for Angular, offering improved performance and a superior developer experience.

Key updates:

hydration is production ready: Teams can now leverage the full capabilities of Hydration in their production environments.
smoother experience: Creating server-side rendering and Hydration-enabled applications is now more streamlined. Developers can directly use the `--ssr` flag with during application setup. This eliminates the need for additional steps to add SSR support later.
enhanced setup process: If the `--ssr` flag is omitted, you will be prompted with its inclusion, ensuring a seamless setup process.

ESM support and performance upgrades: Server builds now support ESM modules, catering to the growing demand for modern JavaScript standards. Furthermore, there are significant improvements in server bundle builds and SSR development server performance.