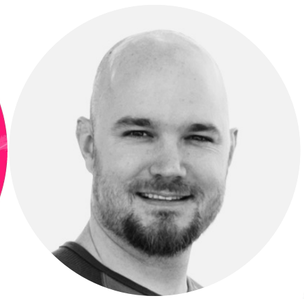
How Styles Work in Angular



Brian Treese
Chief of User Experience

www.socreate.it

HTML

JavaScript

CSS


00:00

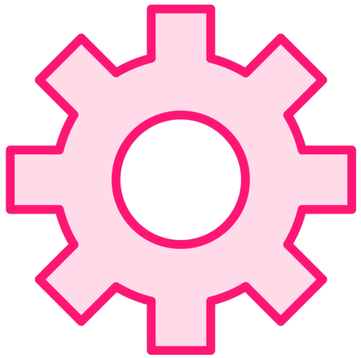
Encapsulation and isolation

Bundle for reuse

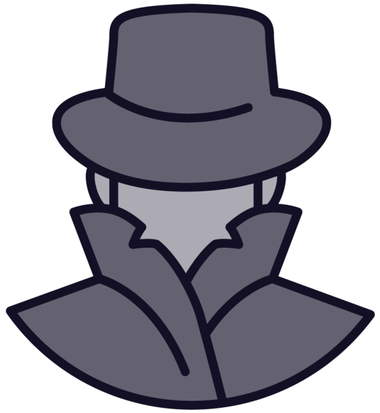
Use in multiple applications

Web Components

Custom elements



HTML templates

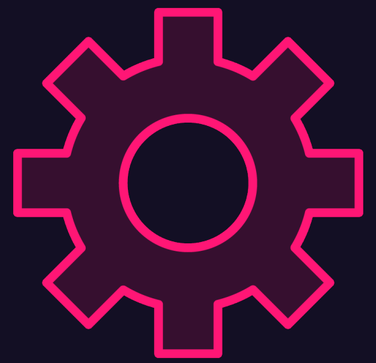


Shadow DOM

```
<my-custom-element>
   Put content and markup here
</my-custom-element>
```

Custom Elements

```
<template id=“template”>
   Put content and markup here
</template>
```



HTML Templates

```
<html>
   <my-custom-element>
      #shadow-root (user-agent)
         Content and markup ends up here
   </my-custom-element>
```

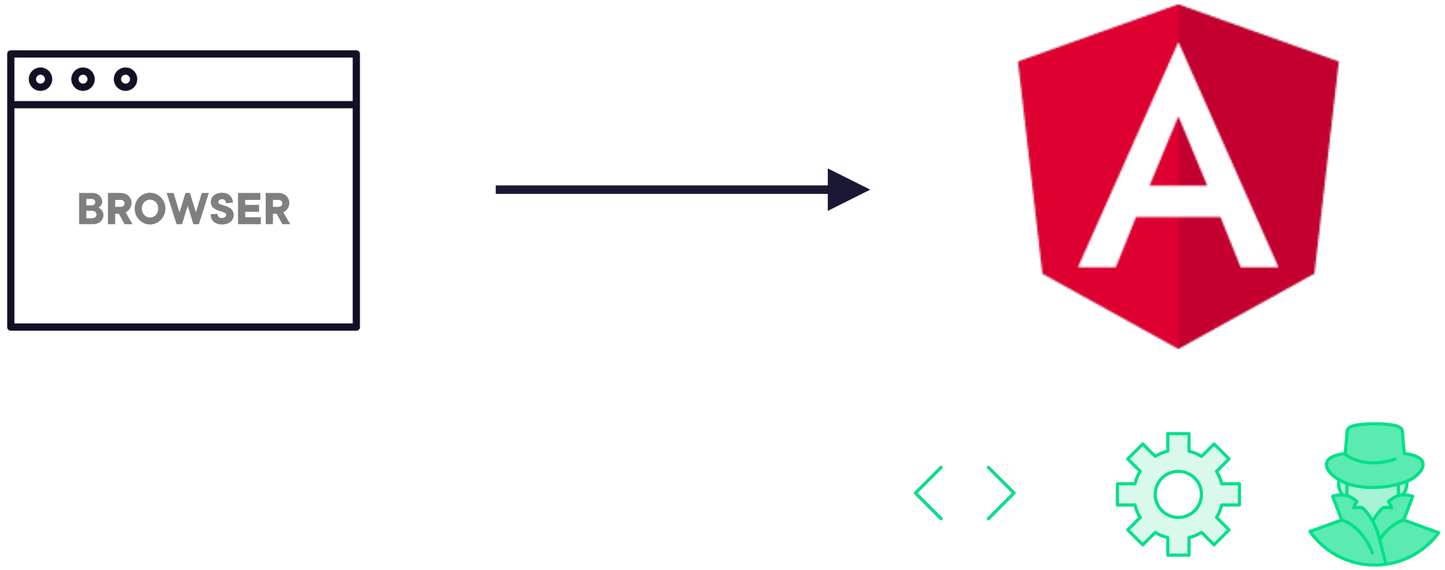
Shadow DOM

Want More on Web Components?

A Practical Guide to Vanilla Web Components

By Leon Revill

Angular Fills Some Gaps



Coming Up...

- Basic component structure
- View encapsulation
- Adding styles
- Scoping CSS selector emulation

Angular Component Structure

Angular Style Guide

https://angular.io/guide/styleguide

View Encapsulation

View Encapsulation Modes

None

Emulated

Shadow DOM

In a Traditional Web Application...

```
<head>
   <head rel="stylesheet" href="/path-to/stylesheet.css">
</head>
```

Don’t Add Styles Tied to These Attributes!

View Encapsulation Modes: None and ShadowDom

This feels wrong

ShadowDom

Native Shadow DOM

Emulated Mode

Component Styles

Other Ways to Add Styles

Embedded

<link>

@import

Inline

styleUrls

Styles in the Template

Linking from an External File

CSS Imports

Inline Styles

styleUrls

~~<link rel="stylesheet" href="/path-to/stylesheet.css">~~

```
styleUrls: ['stylesheet.css'];
```

Syntax Highlighting and Code Completion

Syntax Highlighting and Code Completion

Syntax Highlighting and Code Completion

The Same but Different

styleUrls:

```
styleUrls: [‘stylesheet.css’];
```

Link:

```
<link rel=“stylesheet” href=“/pathto/stylesheet.css”>
```

templateUrl

CSS Preprocessors

CSS Preprocessors

Emulated CSS Selectors

CSS Scoping

:host

:host-context

::ng-deep

“Host”

```
<saa-app-nav>
    <ul>...</ul>
</saa-app-nav>
```

“Shadow Tree”

```
<saa-app-nav>
   <ul>...</ul>
</saa-app-nav>
```

:host

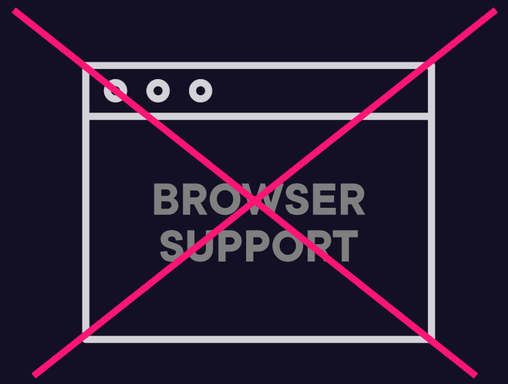
```
:host {
   background: #2A9FBC;
   border-radius: 0.5em;
   margin: 1.5em 0;
   padding: 0;
}
```

:host-context

```
<div class="container">
    <saa-app-nav>
        <ul>...</ul>
    </saa-app-nav>
</div>
```

:host-context

```
:host-context(.container) {
    background: #2A9FBC;
    border-radius: 0.5em;
    margin: 1.5em 0;
    padding: 0;
}
```


BROWSER
SUPPORT

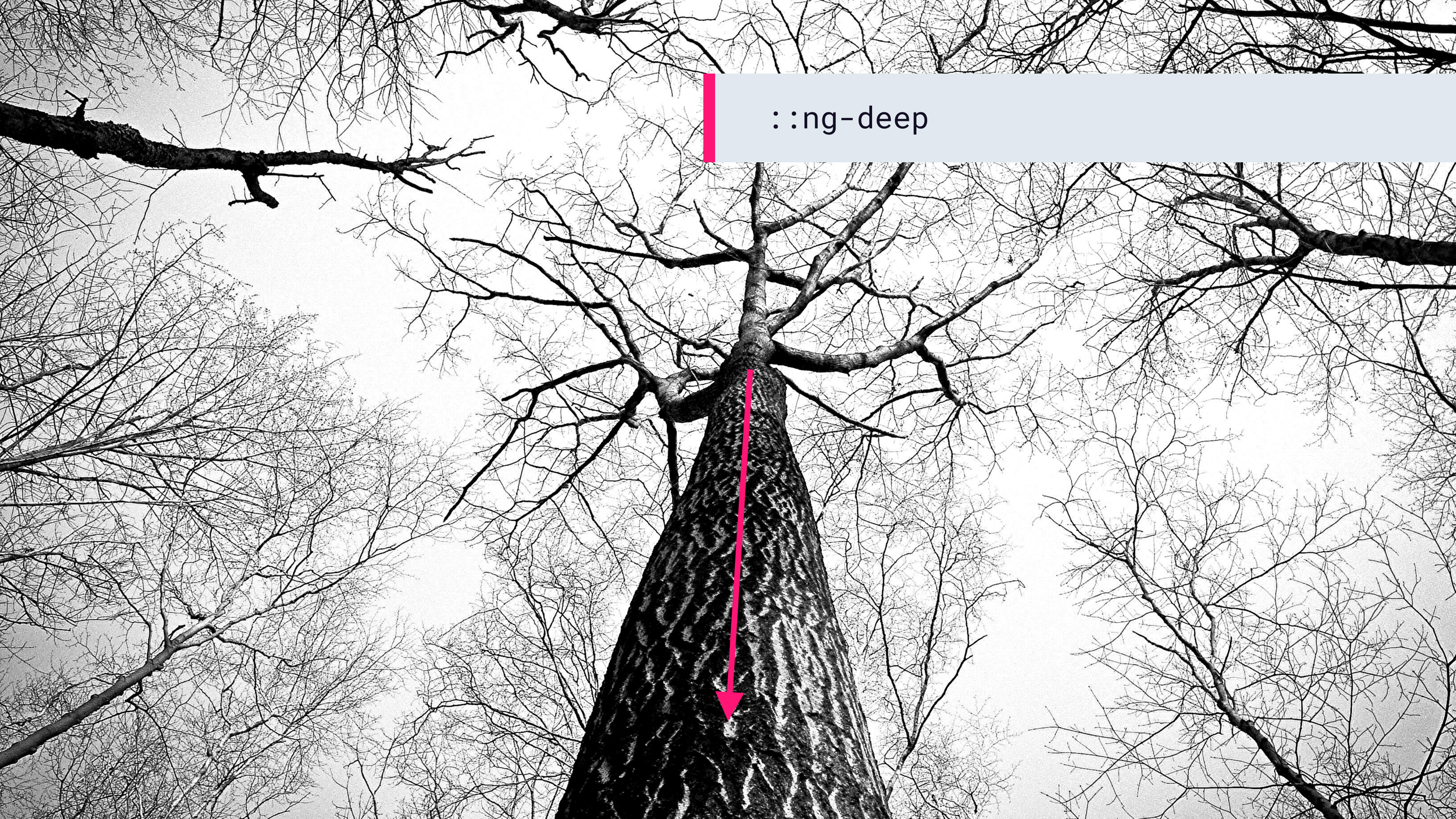
Why Use `:host`?

Why Not `:host-context?`

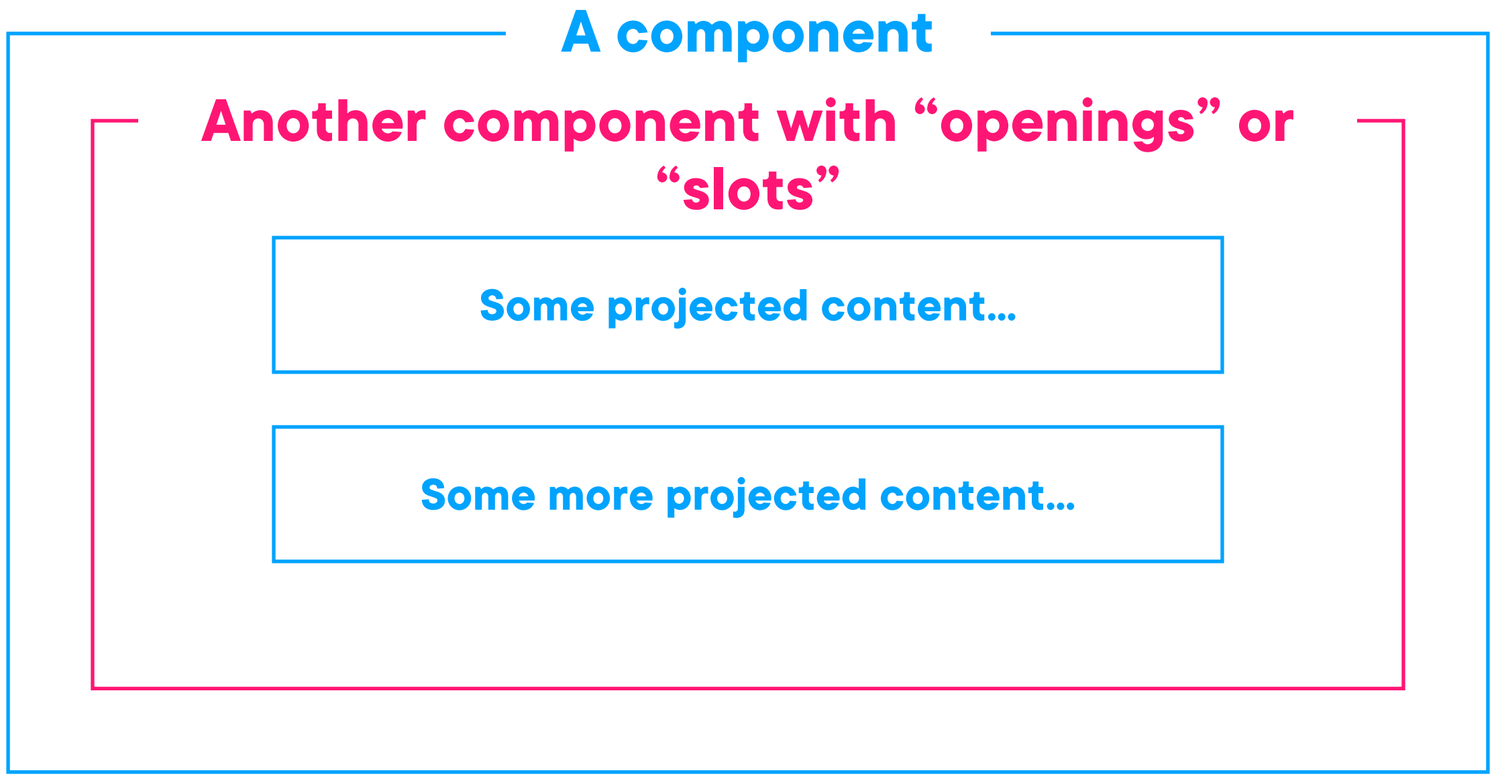
We don’t always know!

`:host-context`

Use it purposefully!


::ng-deep

Content Projection



Content Projection

```
<div>
     This is where content goes
</div>
```

Content Projection

```
<my-component>
    This is also where content goes
</my-component>
```

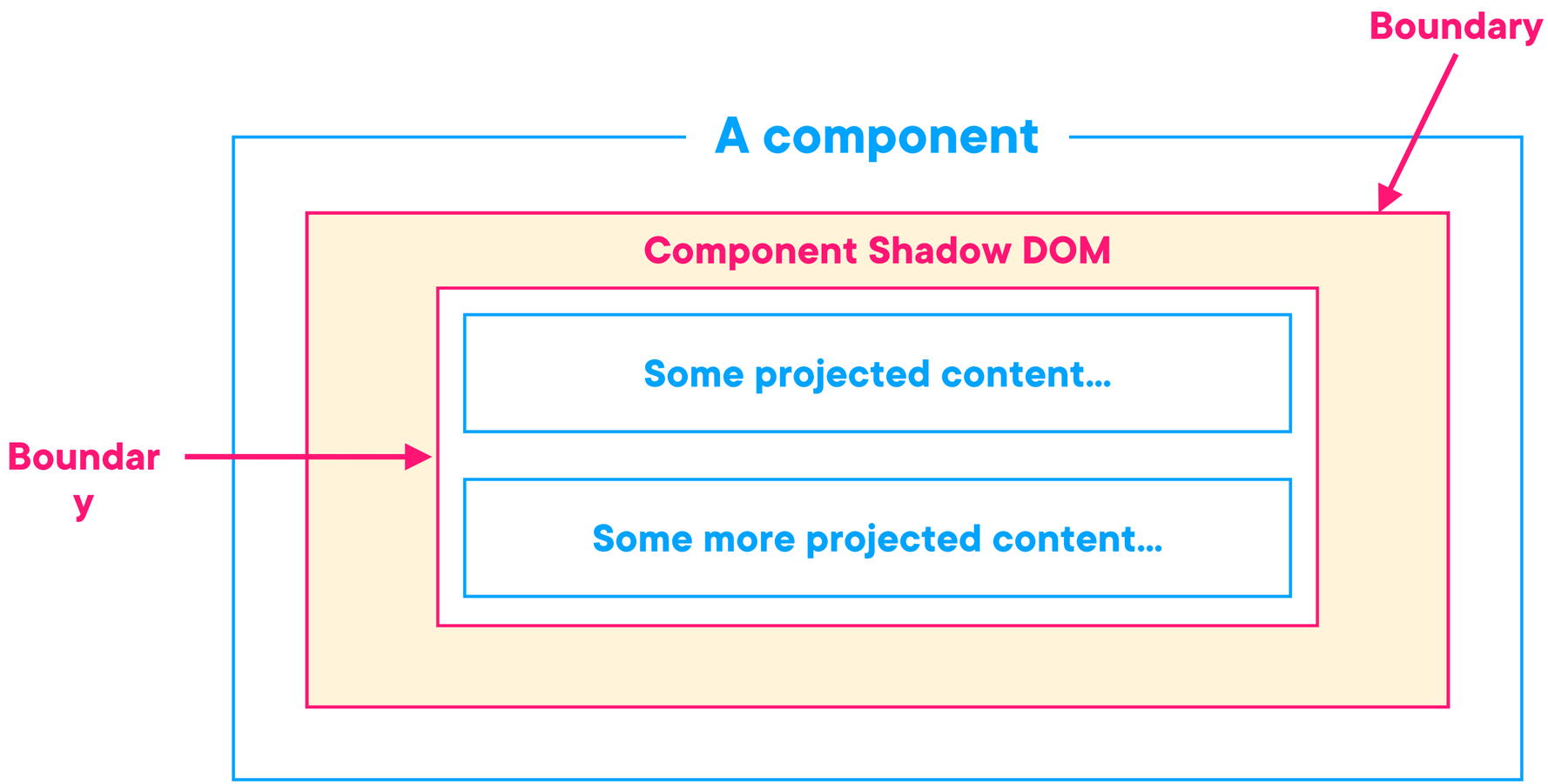
Pop Up Dialog

We Don’t Know

Pop Up Dialog

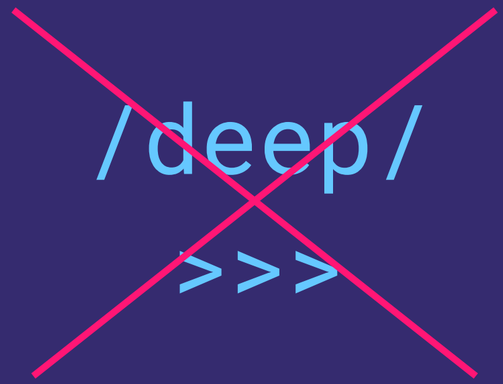
We Do Know

Shadow Boundary




Yikes!

CSS Scoping Module



Deprecated

Where We've Been...

Styling Angular components

Shadow DOM and
view encapsulation

Where We're Headed...

Wrap up how styles work

Create a scalable and maintainable system

Summary

Style encapsulation

Angular Shadow DOM/CSS scoping module emulation

Custom elements

Component HTML templates

View encapsulation: Emulated, ShadowDom, and None

Adding styles: styles property, embedded, linked, imported, inline, external

Using preprocessors

:host, :host-context, ::ng-deep

We need to rethink...

How we craft and organize styles

How we style for predictability, scale, and maintainability