Introduction to Components

Deborah Kurata
Consultant | Speaker | Author | MVP | GDE

@deborahkurata

Module Overview

What is a component?

Creating the component class

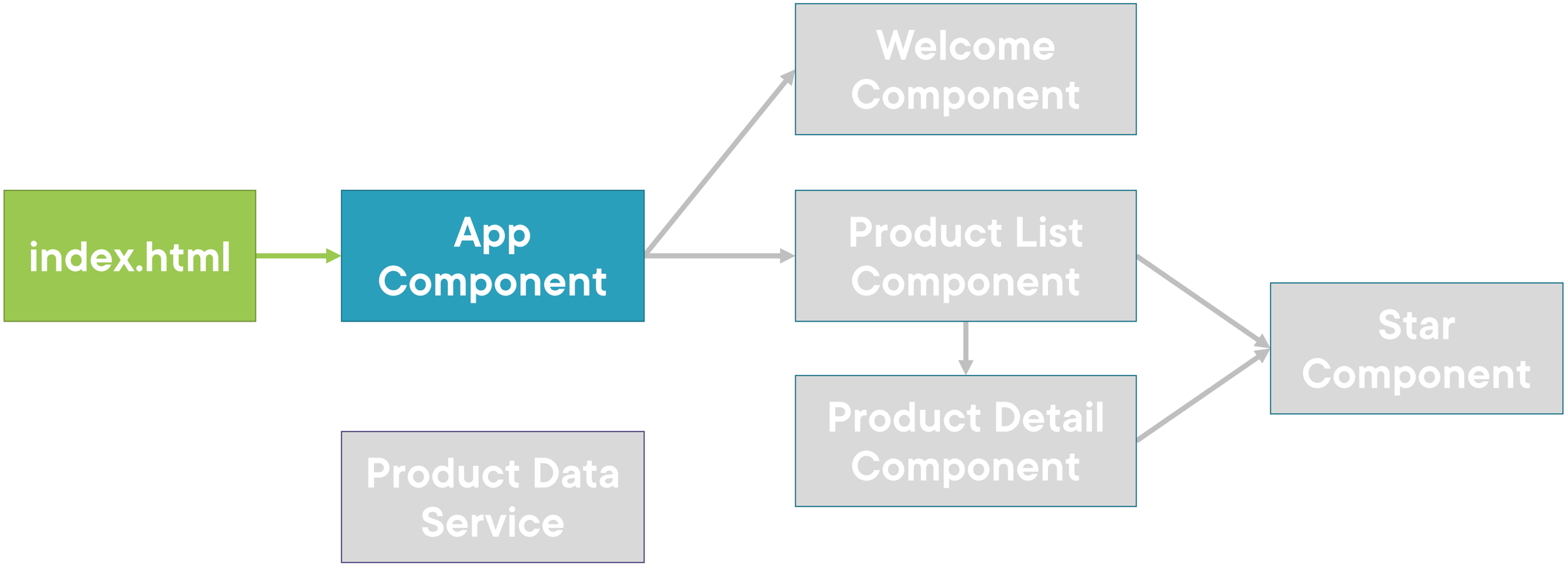
Defining the metadata with a decorator

Importing what we need

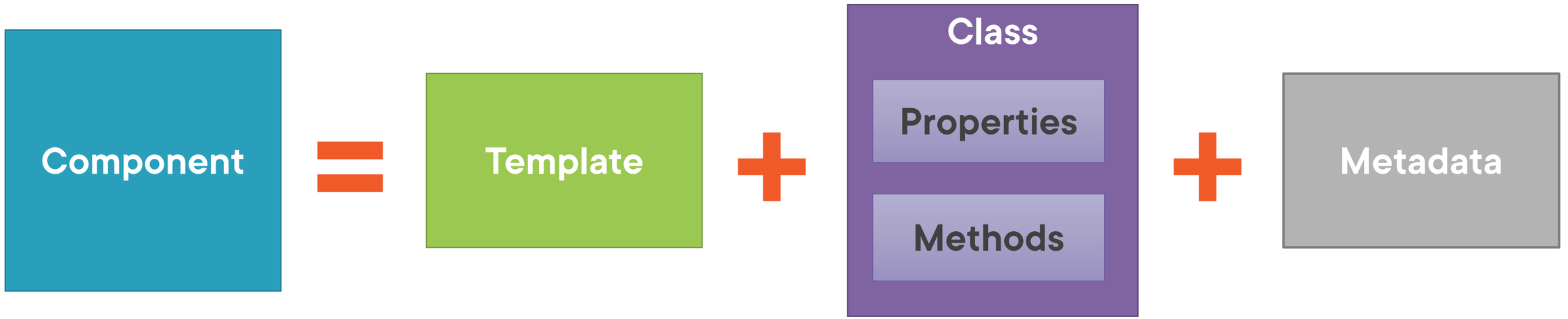
Bootstrapping our app component

Something's wrong!

Application Architecture



What Is a Component?



- View layout
- Created with HTML
- Includes binding and directives

- Code supporting the view
- Created with TypeScript
- Properties: data
- Methods: logic

- Extra data for Angular
- Defined with a decorator

Component

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```

Import

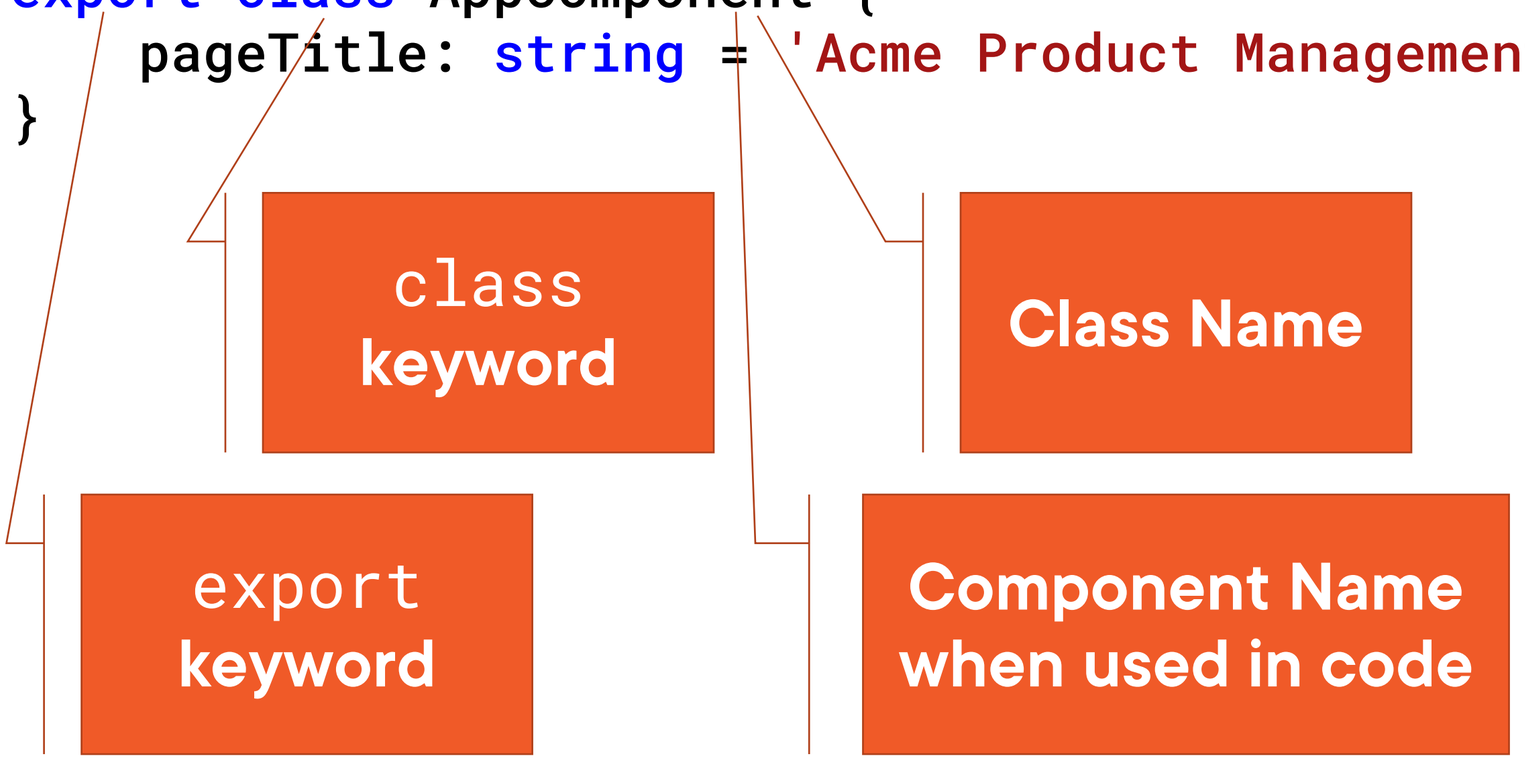
Metadata & Template

Class

Creating the Component Class

app.component.ts

```
export class AppComponent {
    pageTitle: string = 'Acme Product Management';
}
```



Creating the Component Class

app.component.ts

```
export class AppComponent {
    pageTitle: string = 'Acme Product Management';
}
```

Defining the Metadata

app.component.ts

```
@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```

Decorator

A function that adds metadata to a class, its members, or its method arguments.

Prefixed with an @.

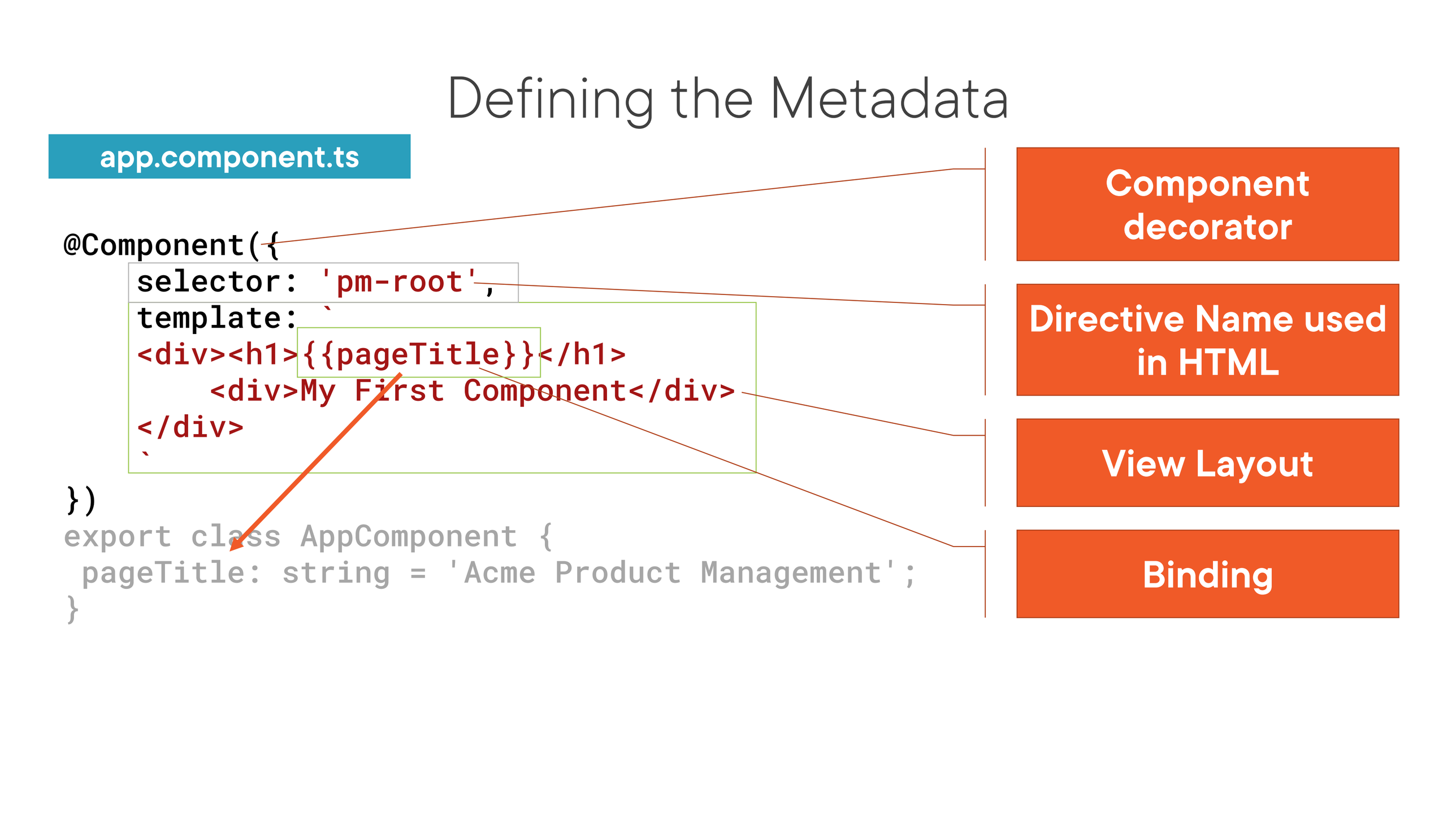
Angular provides built-in decorators.

```
@Component()
```

Defining the Metadata

app.component.ts

```
@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```



Importing What We Need

Before we use an external function or class, we define where to find it

`import` statement

Allows us to use exported members from:

- **Other files in our application**
- **Angular framework**
- **External JavaScript libraries**

Importing What We Need

app.component.ts

```
@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```

Importing What We Need

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```

import keyword

Angular library name

Member name

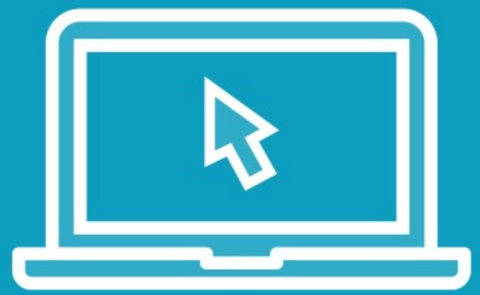
Completed Component

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```

Demo



Creating the App component

Bootstrapping Our App Component

Host the application

Defining the Angular module

Web Browser

Web Server

URL Request
(www.mysite.com)

Response

index.html

JavaScript

Single Page Application (SPA)

`index.html` contains the main page for the application

This is often the only Web page of the application

Hence an Angular application is often called a Single Page Application (SPA)

Hosting the Application

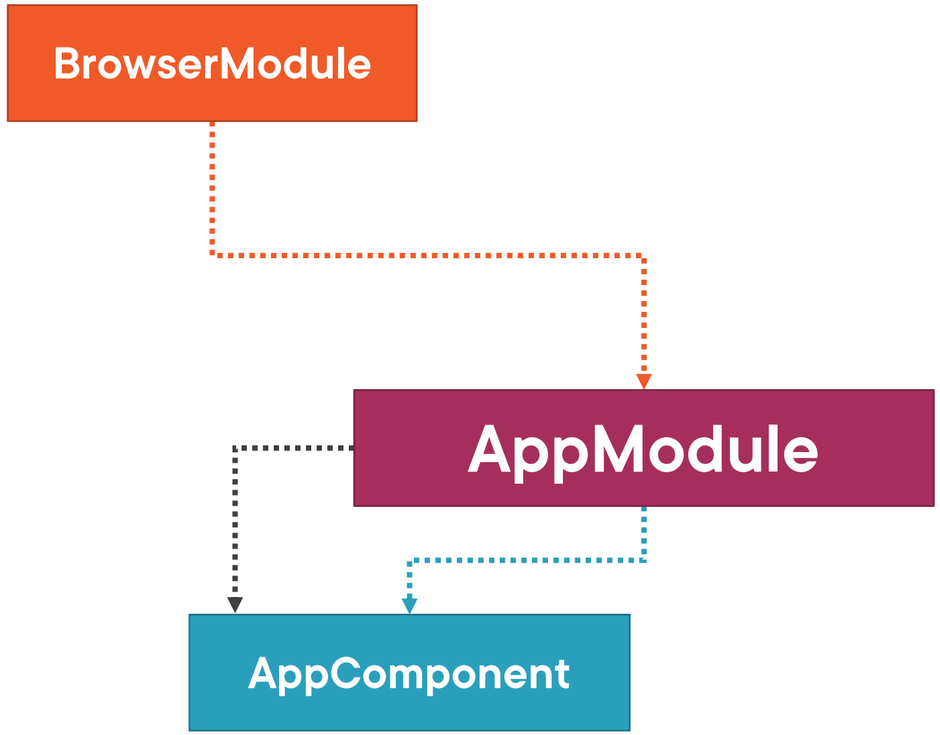
index.html

```
<body>
 <pm-root></pm-root>
</body>
```

app.component.ts

```
import { Component } from '@angular/core';

@Component({
    selector: 'pm-root',
    template: `
    <div><h1>{{pageTitle}}</h1>
        <div>My First Component</div>
    </div>
    `
})
export class AppComponent {
 pageTitle: string = 'Acme Product Management';
}
```



Organization

Boundaries

Template resolution environment

Imports

Exports

Declarations

Bootstrap

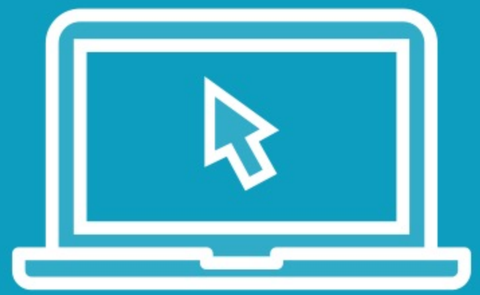
Defining the Angular Module

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent }  from './app.component';

@NgModule({
  imports: [ BrowserModule ],
  declarations: [ AppComponent ],
  bootstrap: [ AppComponent ]
})
export class AppModule { }
```

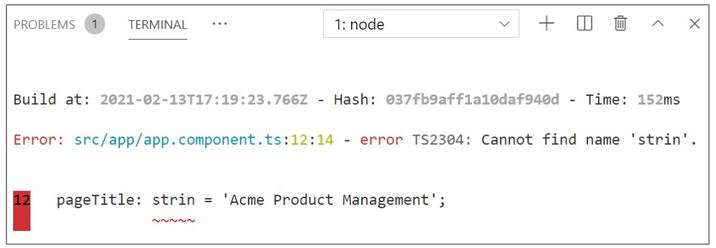
Demo



Bootstrapping our App component

Angular compiles our
HTML templates and
TypeScript components
to JavaScript

Something's Wrong!




```
export class AppComponent {
 pageTitle = 'Acme PM';
 PageTitle = 'Something else';
}
```

```
@NgModule({
  declarations: [
    AppComponent,
    ProductListComponent
  ],
  imports: […],
  bootstrap: […]
})
export class AppModule { }
```

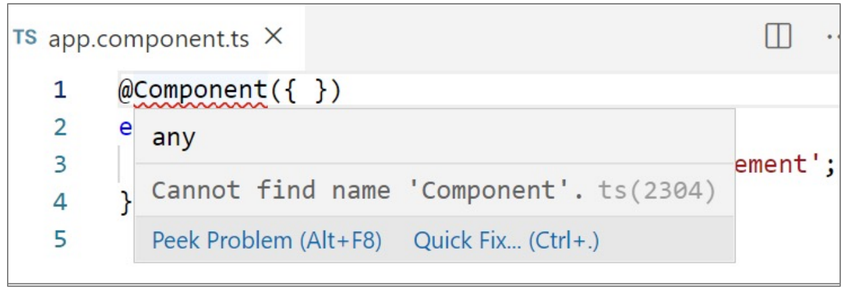
Compiler displays syntax errors

Casing matters

Components must be declared in an Angular module

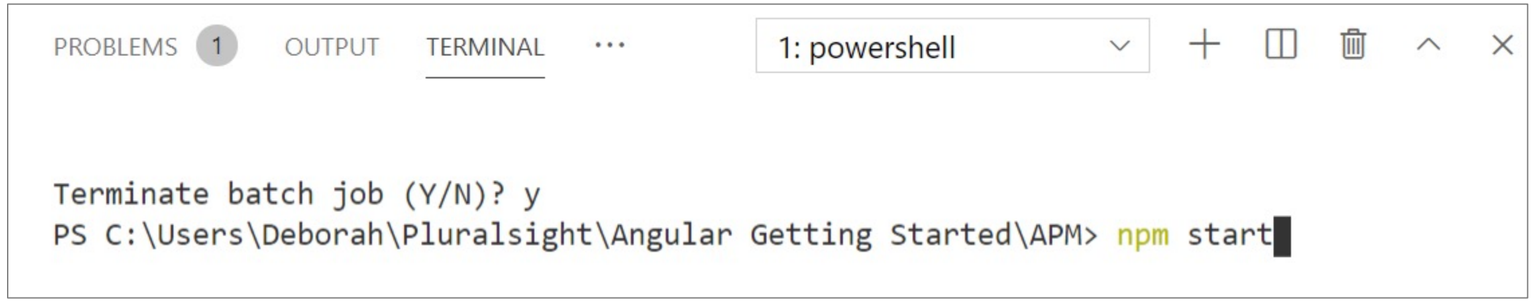
Start with Your Code Editor

Check for squiggly lines

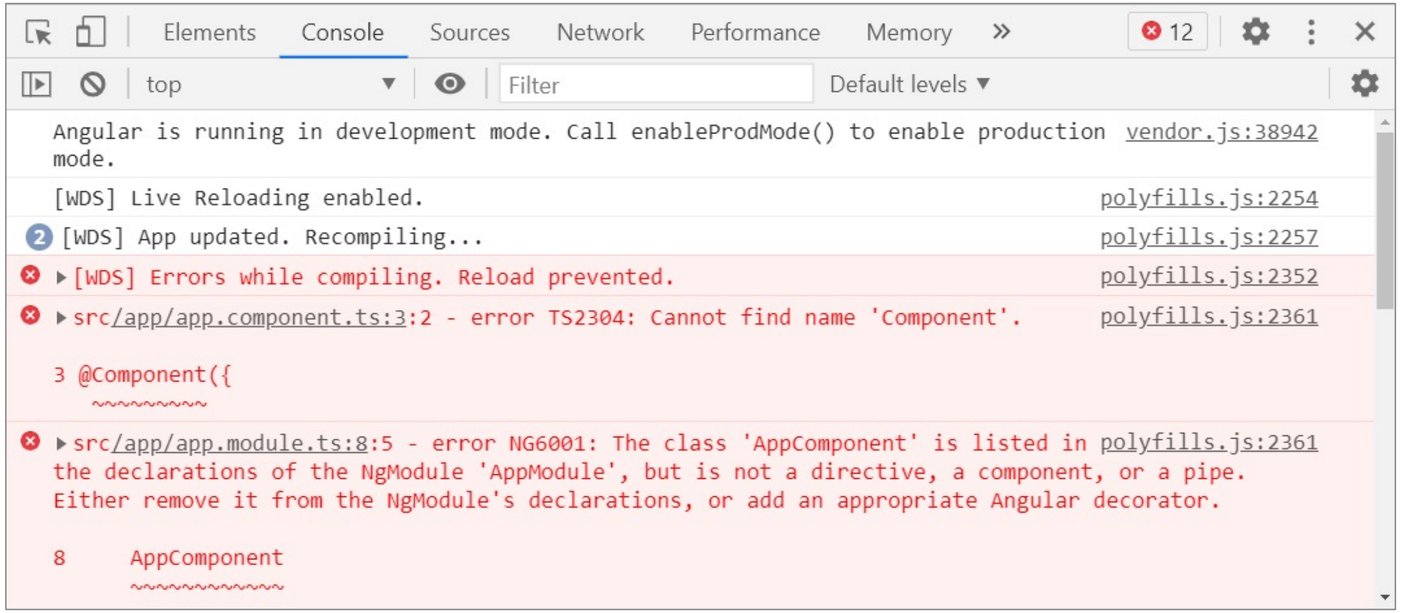


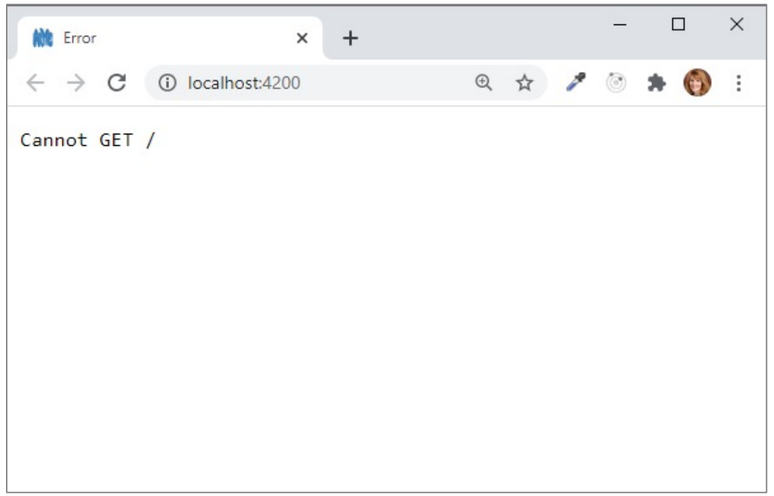
Open the terminal

Stop (Ctrl+C) and Restart



Use the browser's developer tools

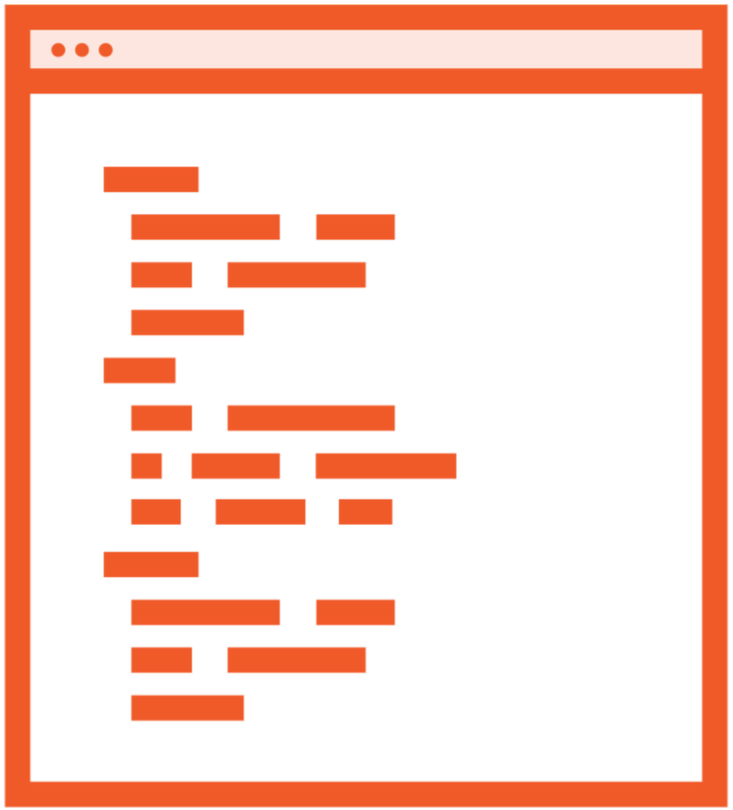




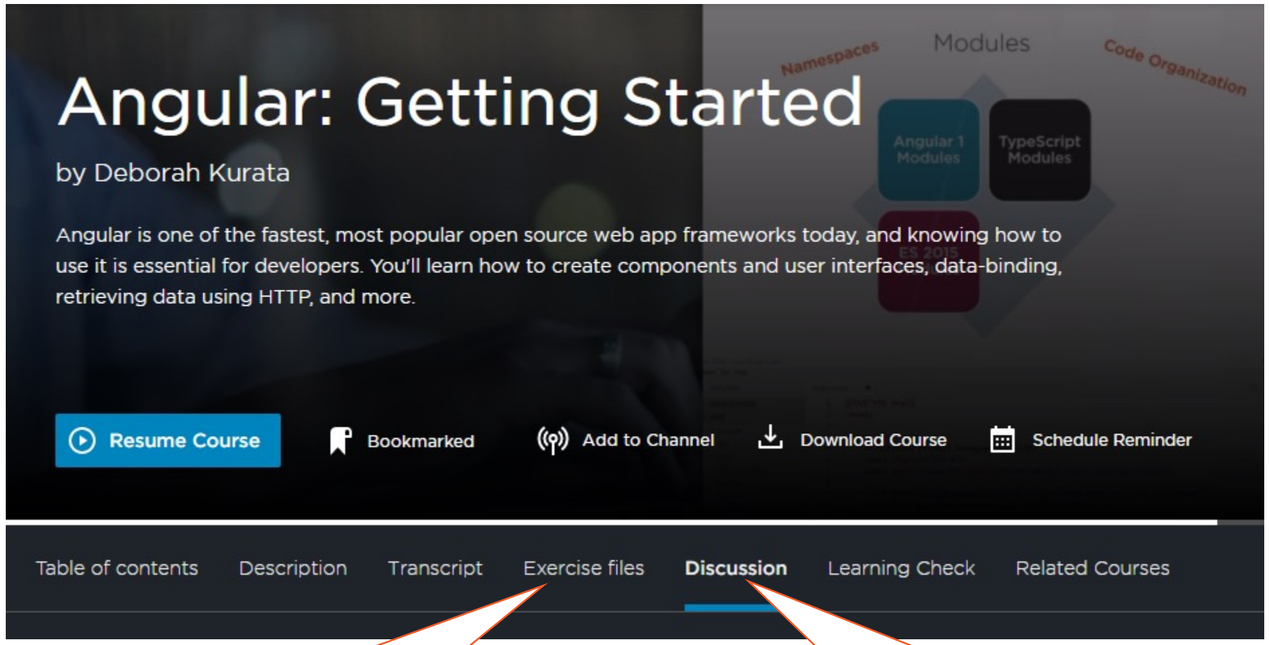
Often means an error prevented the compiler from completing

Use the VS Code terminal to view the errors

Recheck Your Code



- HTML
  - Close tags
  - Angular directives are case sensitive
- TypeScript
  - Close braces
  - TypeScript is case sensitive



- Code “as of” the end of each module
- Copy of the slides

- Review posted issues
- Post your own issues

Component Checklist

Class -> Code

Decorator -> Template and Metadata

Import what we need

Component Checklist:
Class

Clear name

- **Use PascalCasing**
- **Append "Component" to the name**

`export` keyword

```
export class AppComponent {
  pageTitle: string = 'Acme Product Management';
}
```

Component Checklist: Members

Data in properties

- Appropriate data type
- Appropriate default value
- camelCase with first letter lowercase

Logic in methods

- camelCase with first letter lowercase

```
export class AppComponent {
  showImage: boolean =  false;    // property

  toggleImage(): void {           // method
    this.showImage = true;
  }
}
```

Component Checklist: Metadata

Component decorator

- Prefix with @; Suffix with ()

`selector`: Component name in HTML

- Prefix for clarity

`template`: View's HTML

- Correct HTML syntax

```
@Component({
  selector: 'pm-root',
  template: `
    <div><h1>{{pageTitle}}</h1></div>
  `
})
```

Component Checklist: Import Statement

Defines where to find the members that this component needs

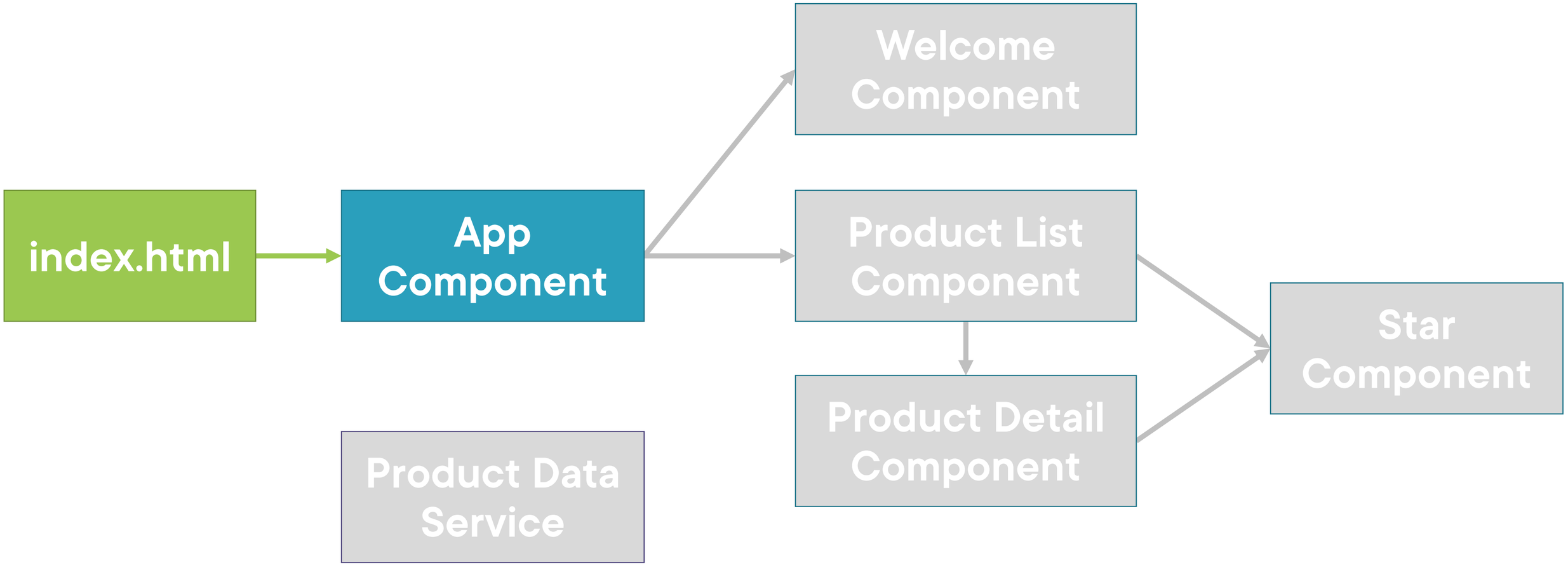
`import` keyword

Member name
- Correct spelling/casing

Path
- Enclose in quotes
- Correct spelling/casing

```
import { Component } from '@angular/core';
```

Application Architecture



Coming up next ...

Templates, Interpolation, and Directives