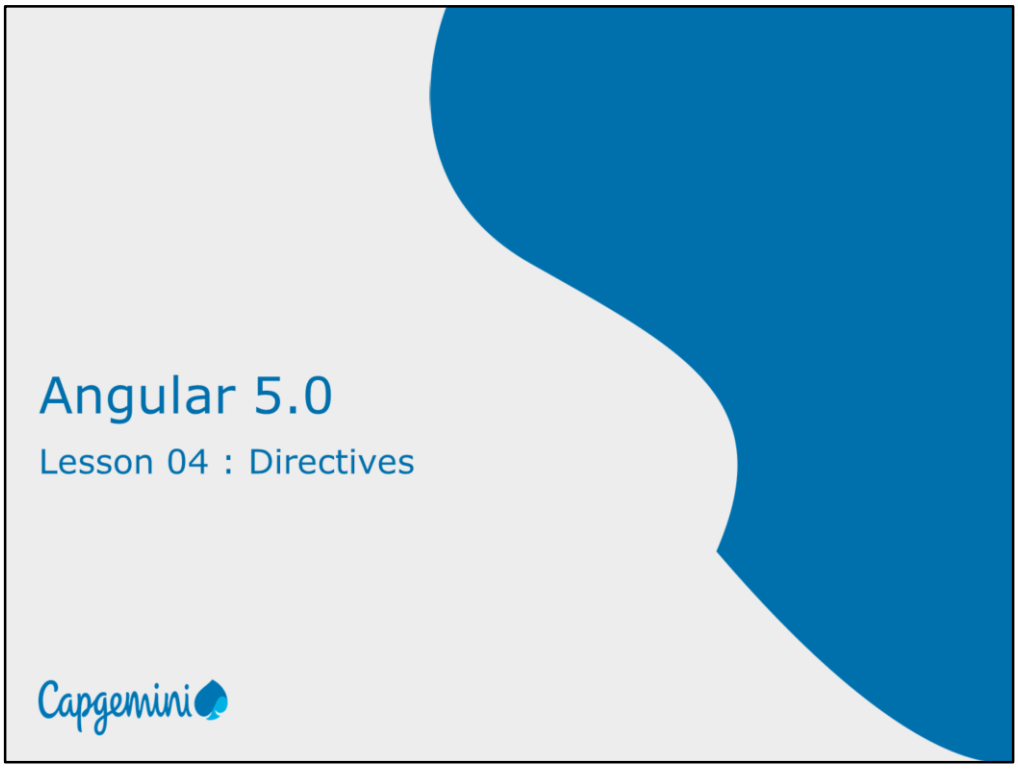
Instructor Notes:

Add instructor notes here.



Instructor Notes:

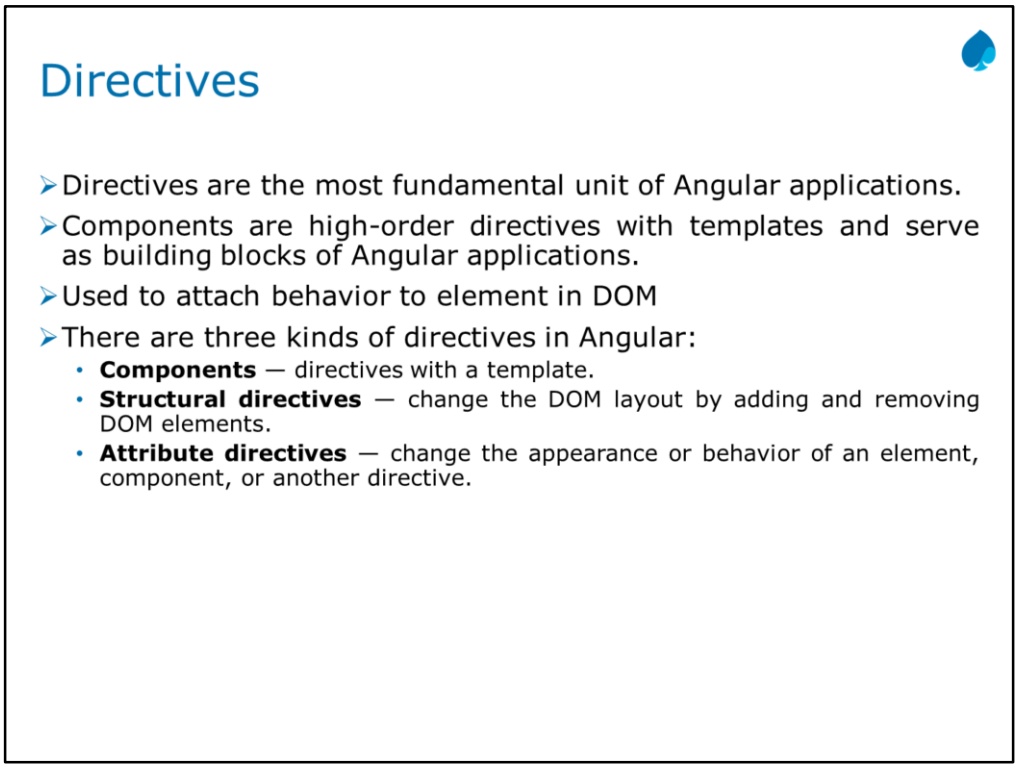
Add instructor notes here.

Lesson Objectives

- What are directives?
- Types of directives - component, structural and attribute
- Creating a basic directive
- Handling event & Binding input in attribute directive
- Creating your own structural directive
- Using the structural directive
- Binding input to a structural directive

Instructor Notes:

Add instructor notes here.



A Component is really a directive with a template. It's the most common of the three directives and we tend to write lots of them as we build applications.

Structural directives can change the DOM layout by adding and removing DOM elements. NgFor and NgIf are two familiar examples.

An Attribute directive can change the appearance or behavior of an element. The built-in NgStyle directive, for example, can change several element styles at the same time.
Angular 2 categorizes directives into 3 parts:
Directives with templates known as **Components**
Directives that creates and destroys DOM elements known as **Structural Directives**
Directives that manipulate DOM by changing behavior and appearance known as **Attribute Directives**

Instructor Notes:

Add instructor notes here.

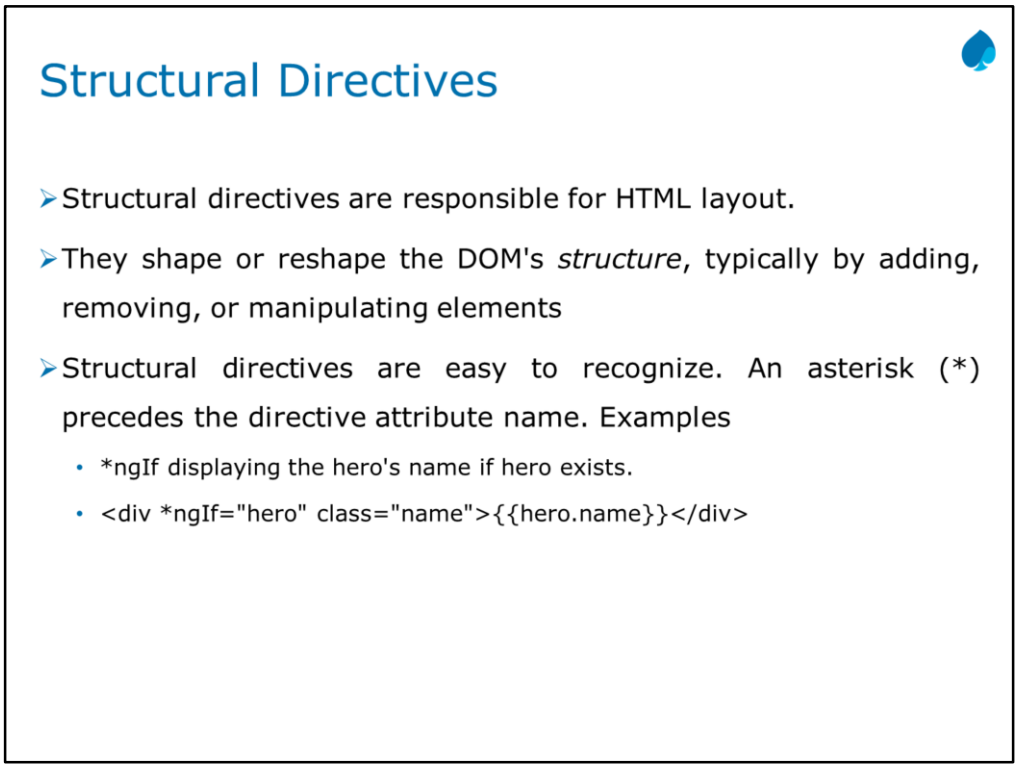
Demo

- Component Directive

Add the notes here.

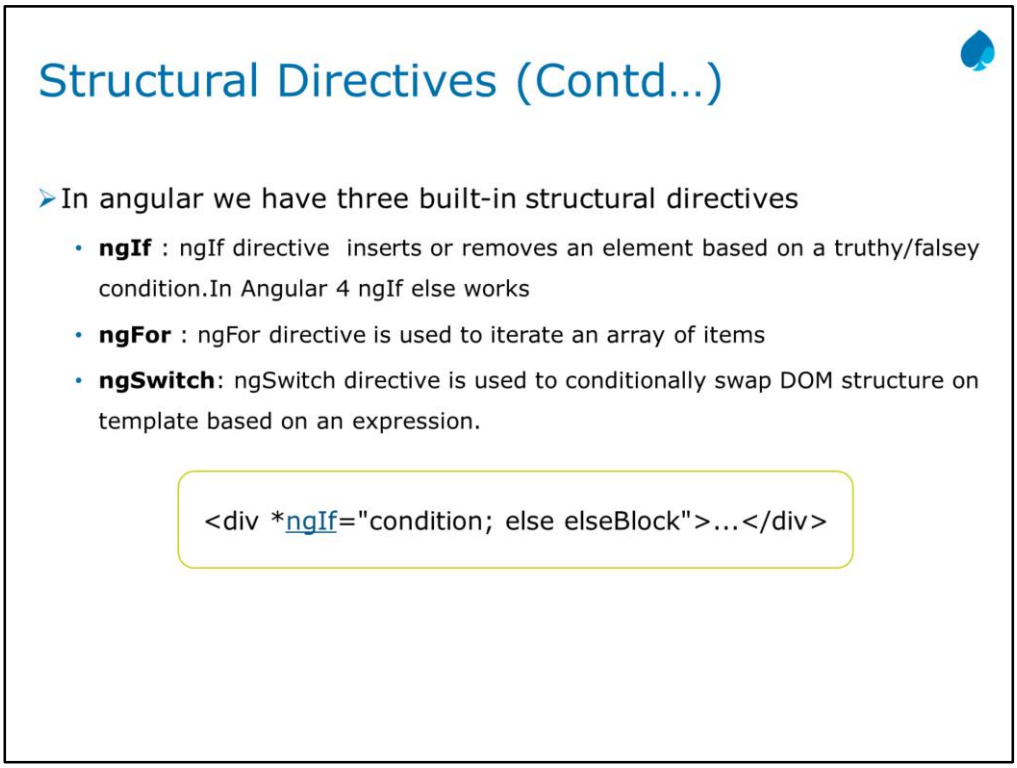
Instructor Notes:

Add instructor notes here.



Structural directives are not DOM-friendly in the sense that they create, destroy, or re-create DOM elements based on certain conditions.
This is a huge difference from what hidden attribute directive does. This is because hidden retains the DOM element but hides it from the user, whereas structural directives like *ngIf destroy the elements.
*ngFor and [ngSwitch] are also common structural directives and you can relate them to the common programming flow tasks.

Instructor Notes:



ngIf

ngIf is a structural directives that removes or recreates a portion of the DOM tree based on an expression. If the expression assigned to the ngIf evaluates to a false value; the element and its children are removed from the DOM, whereas if the expression evaluates to a true value a copy of the element and its children are reinserted into the DOM. ngIf adds or removes an element subtree (an element and its children) in the DOM. In this example, the parent div has zero children when ngIf is false.

ngFor

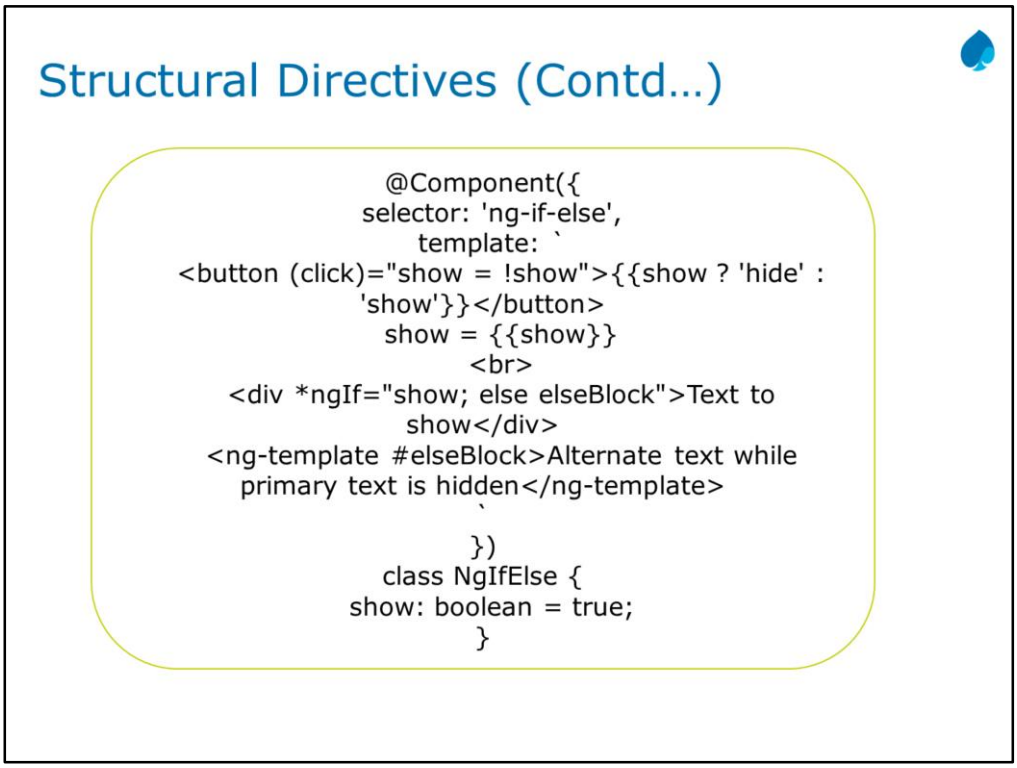
ngFor repeats a portion of the DOM tree once for each item in an iterable list. The let keyword explicitly defines the variable as a local variable used only in the template. It can be referenced on that element or any sibling element or on any of its child element.

Note:
ES 2015 has both for..of loop and for..in loop. for..of loop is used to iterate over an iterable objects such as an array where as for..in loop is used to iterate over the properties of an object.

ngSwitch, NgSwitchWhen & ngSwitchDefault

ngSwitch simply inserts nested elements based on which match expression matches the value obtained from the evaluated switch expression. ngSwitchWhen property is used to inform NgSwitch which element to display when the expression is evaluated. If a matching expression is not found via a ngSwitchWhen property then an element with the ngSwitchDefault attribute is displayed.

Instructor Notes:



ngIf

ngIf is a structural directives that removes or recreates a portion of the DOM tree based on an expression. If the expression assigned to the ngIf evaluates to a false value; the element and its children are removed from the DOM, whereas if the expression evaluates to a true value a copy of the element and its children are reinserted into the DOM. ngIf adds or removes an element subtree (an element and its children) in the DOM. In this example, the parent div has zero children when ngIf is false.

ngFor

ngFor repeats a portion of the DOM tree once for each item in an iterable list. The let keyword explicitly defines the variable as a local variable used only in the template. It can be referenced on that element or any sibling element or on any of its child element.

Note:
ES 2015 has both for..of loop and for..in loop. for..of loop is used to iterate over an iterable objects such as an array where as for..in loop is used to iterate over the properties of an object.

ngSwitch, NgSwitchWhen & ngSwitchDefault

ngSwitch simply inserts nested elements based on which match expression matches the value obtained from the evaluated switch expression. ngSwitchWhen property is used to inform NgSwitch which element to display when the expression is evaluated. If a matching expression is not found via a ngSwitchWhen property then an element with the ngSwitchDefault attribute is displayed.

Instructor Notes:

Add instructor notes here.

Demo

- Demo ngIf Directive
- Demo ngFor Directive
- Demo ngSwitch Directive

Add the notes here.

Instructor Notes:

Creating own structural directive

- We can create custom attribute directives and custom structural directives using @Directive decorator.
- Structural directives are responsible for HTML layout.
- We can add and remove elements in DOM layout dynamically.
- The HTML element using directive is called host element for that directive.
- To add and remove host elements from DOM layout we can use TemplateRef and ViewContainerRef classes in our structural directive.
- To change DOM layout we should use TemplateRef and ViewContainerRef in our structural directive.
- **TemplateRef** : It represents an embedded template that can be used to instantiate embedded views.
- **ViewContainerRef :** It represents a container where one or more views can be attached.

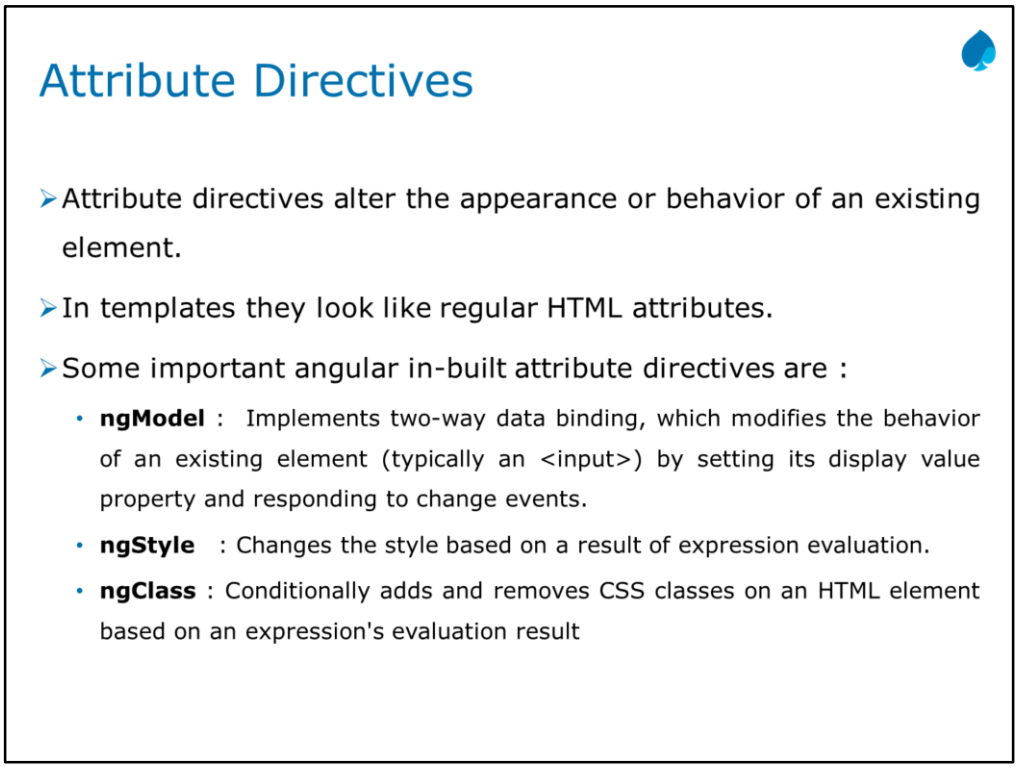
Instructor Notes:

Demo

- Demo Own structural Directive

Instructor Notes:

Add instructor notes here.



Attribute directives, as the name goes, are applied as attributes to elements. They are used to manipulate the DOM in all kinds of different ways except creating or destroying them. I like to call them DOM-friendly directives.
Directives in this categories can help us achieve one of the following tasks:
Apply conditional styles and classes to elements
<p [style.color]="'blue'">Directives are awesome</p> **Hide and show elements, conditionally (different from creating and destroying elements)**
<p [hidden]="shouldHide">Directives are awesome</p> **Dynamically changing the behavior of a component based on a changing property**

Instructor Notes:

Add instructor notes here.

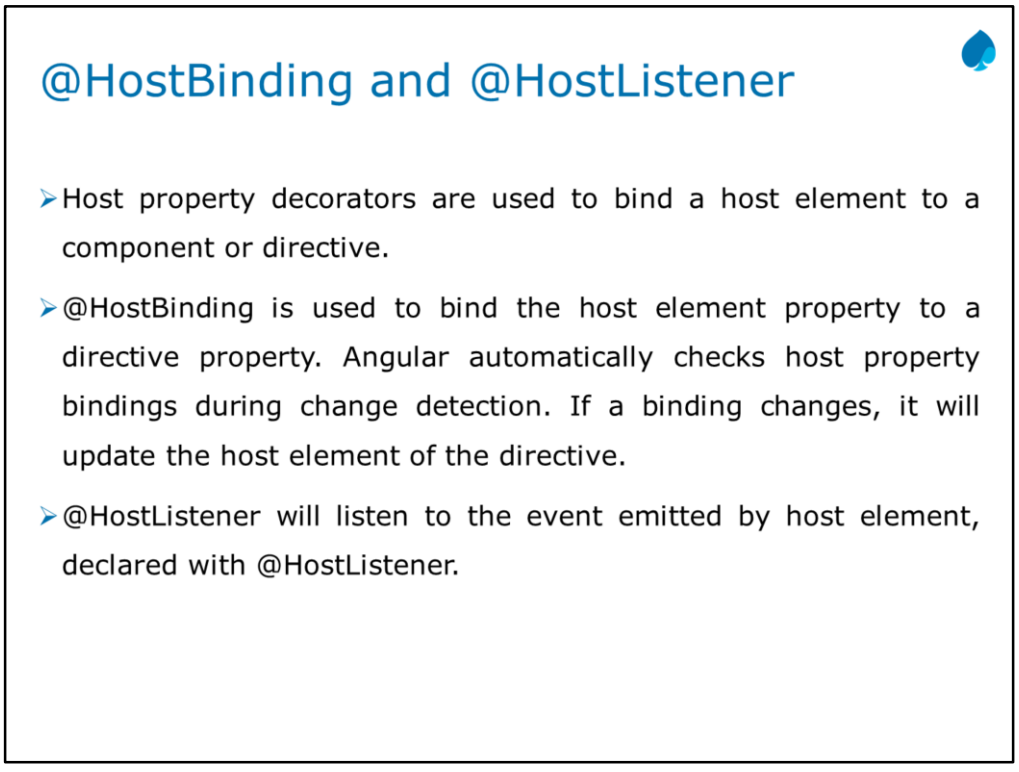
Demo

- Demo ngStyle Directive
- Demo ngClass Directive

Add the notes here.

Instructor Notes:

Add instructor notes here.



HostListener - Declares a host listener. Angular will invoke the decorated method when the host element emits the specified event.

#HostListener - will listen to the event emitted by host element, declared with @HostListener.

HostBinding -Declares a host property binding.Angular automatically checks host property bindings during change detection. If a binding changes, it will update the host element of the directive.

#HostBinding - will bind property to host element, If a binding changes, HostBinding will update the host element.

Host Listeners *are event listeners attached to any element that hosts (the directive is placed on) the directive.*

Instructor Notes:

Add instructor notes here.

Demo

- Demo Custom Directive

Add the notes here.

Instructor Notes:

Add instructor notes here.

Summary

- Component Directives is a directive with a template.
- Directives can't be bootstrapped
- Structural directives change the DOM layout by adding and removing DOM elements.
- Attribute directives changes the appearance or behavior of a DOM element.
- In order to use *ngModel* in the application components, we need to compulsorily add FormsModule in the Imports array of Application Module class
- *ngNonBindable* tells the Angular not to compile or bind a particular section of a DOM.
- Using *ngStyle* directive we can set CSS properties for the DOM element from Angular expressions
- *ngClass* directive allows us to dynamically set and change the CSS classes for a given DOM element

Instructor Notes:

Add instructor notes here.

Lab

- Lab 2

Add the notes here.