Angular 16 Online Training

sahosoft
Online Learning Platform

05

By: Chandan Kumar

sahosoft
Online Learning Platform

Encapsulation

Angular Components

The Angular components are consist of three things:

- Component class
- Template
- Style

The combination of these three factors makes an angular component reusable in an application. In theory, when you create a component, you create a Web component in some way (the angular components are not Web components) to take advantage of the Shadow DOM. You can also use Angular with browsers, which are not compatible with Shadow DOM because Angular has its own emulation and can emulate Shadow DOM

Encapsulation

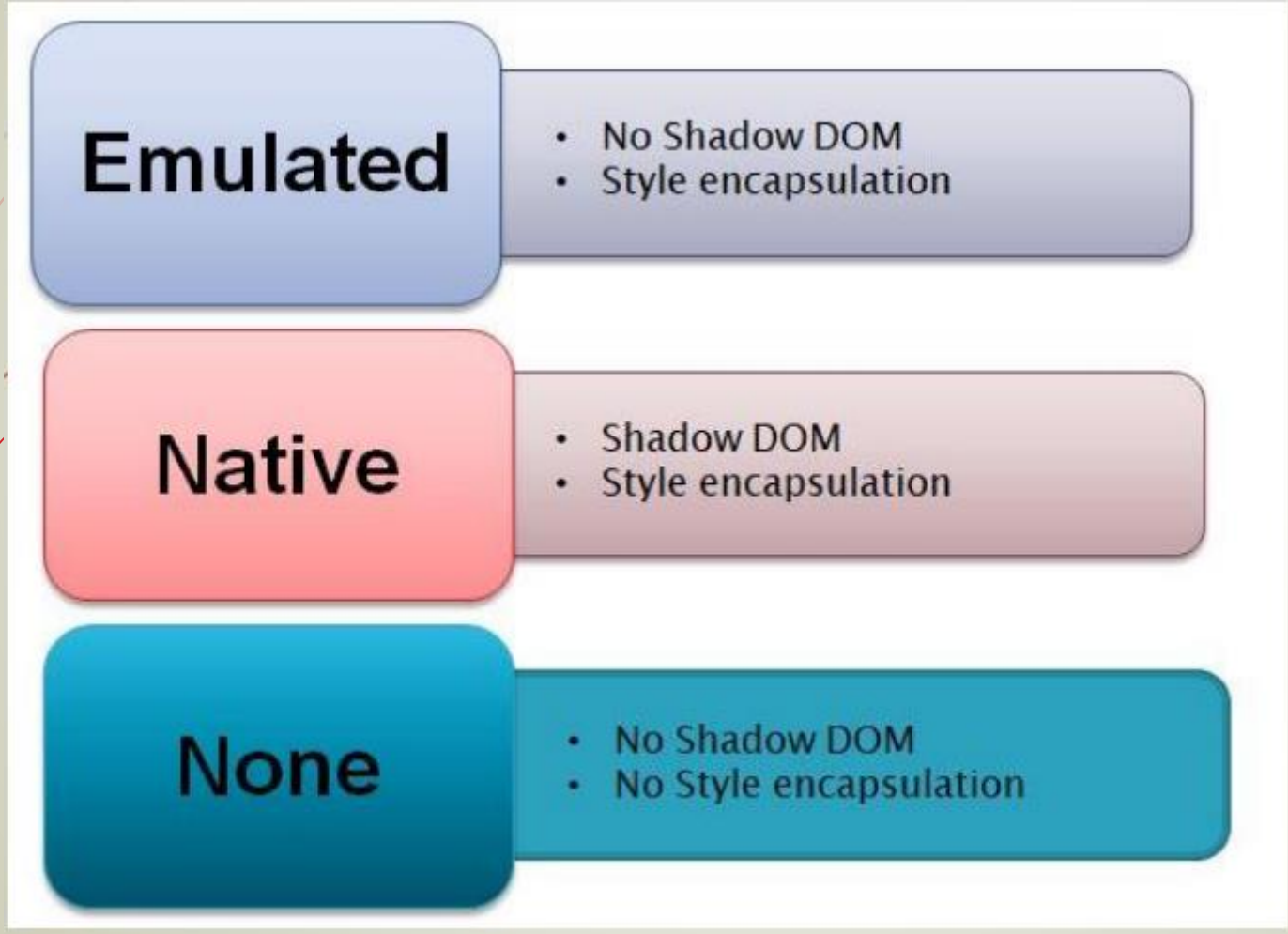
To emulate the shadow DOM and encapsulate styles, Angular provides three types of view encapsulation. Are the following:

Emulated (default): The main HTML styles are propagated to the component. The styles defined in the component's @Component decorator are limited to this component only.

Native/shadow dom: The main HTML styles are not propagated to the component. The styles defined in the component's @Component decorator are limited to this component only.

None: the component styles are propagated to the main HTML and therefore are visible to all components of the page. with apps that have None and Native components in the application.. All components with encapsulation None will have their duplicate styles on all components with native encapsulation

Encapsulation



Input Property

Input property is used within one component (child component) to receive a value from another component (parent component). This is a one-way communication from parent to child. A component can receive a value from another component through component input property.

```
@Component({
      selector: 'app-child',
       templateUrl: './child.component.html',
       styleUrls: ['./child.component.css']
       inputs: [isparentdata: PData]
})
export class ChildComponent {
  pData = false;
}
```

Output Property

Output property is used send data from one component (child component) to calling component (parent component). This is a one-way communication from child to parent. This property name becomes a custom event name for the calling component. The output property can also create an alias with the property name as Output (alias) and now this alias name will be used in the custom event link in the call component. Now we will see how to use the output property. It is the topic of the Component Interaction in angular. As we know, the angular application is based on small components, so it is very difficult to pass data from a child component to a parent component, in this scenario the output property is very useful. The angular components have a better way of notifying the parent components that something has changed through events. The "outputs" identify events that a component can fire to send information from the hierarchy to its parent from its child component.

```
@Component({
       selector: 'app-child',
        templateUrl: './child.component.html',
        styleUrls: ['./child.component.css']
        outputs: [ischildEvent: childEvent]
})
export class ChildComponent {
  childEvent = new EventEmitter();
}
```