Welcome in day 4 workshop,

1. Encapsulation (parent to child css shaing)
2. Input & Output Property
3. Providers property

How to create a parent and child communication

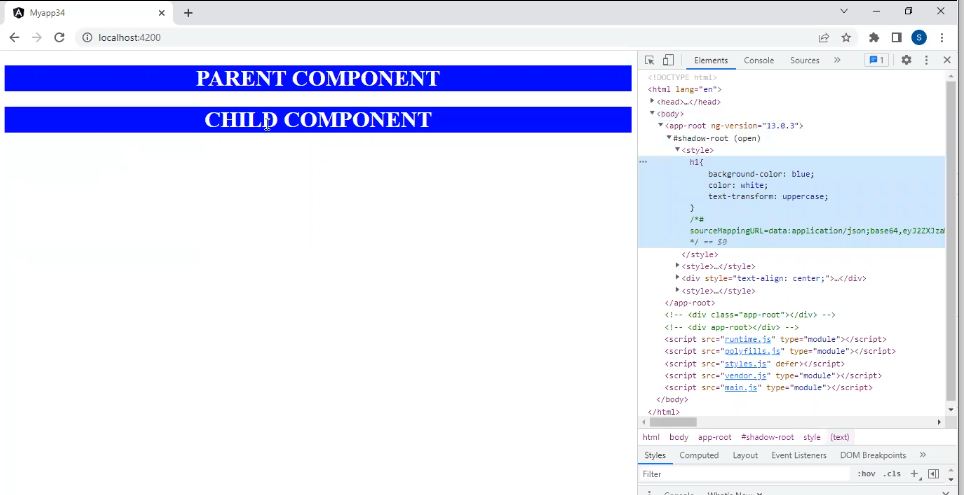
Using Input and output

B selector in A then A is parent of B

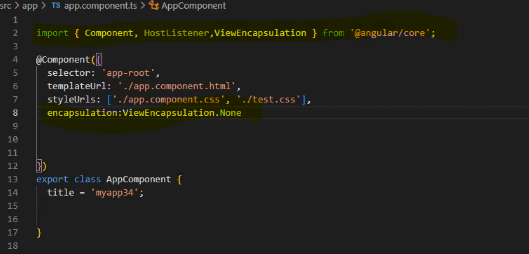
Parent css will only applied to child and for which only encapsulation is created

3 Types of Encapsulation

1. Emulated(default)-No shadow Dom(CSS of parent will not go to child), style encapsulation
2. Native – Shadow DOM(CSS of parent will go to child), Style encapsulation
3. None- No shadow DOM, No style encapsulation



Here as we can see after using parent in Metadata elements we can see entire code in Dom structure of our code and we can import it



How to use import & encapsulation

encapsulation: ViewEncapsulation.None-> Will go to child to child and parent

encapsulation: ViewEncapsulation.ShadowDon-> Only pass to its child

encapsulation: ViewEncapsulation.Native-> Not pass to anyone

Activity

How many metadata’s we have previously seen?

If Parent is also none and child none then what will happen?

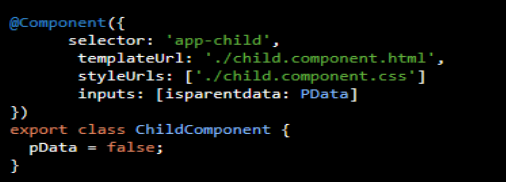
Parent first applied then child will apply none then childs one will be applied

Local css always override global css

Input and Output Property

Input Property

When Parent to child data transfer



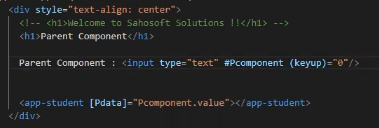
Output

When child to parent data transfer



How to use Input

1. Write a method in parent component

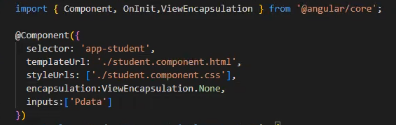


So we want this input data to reflected in child so,

When created one variable of #Pcomponent and created event for button release

(keyup)= ‘0’

Pdata will transfter date

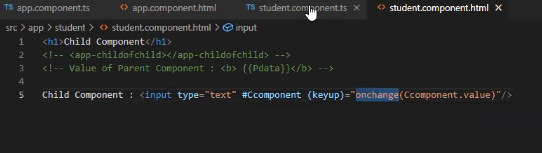


In html we will reflect data using {{Pdata}}. So data will be reflected into it.

Why do we use Event: Just to take html changes on ts file

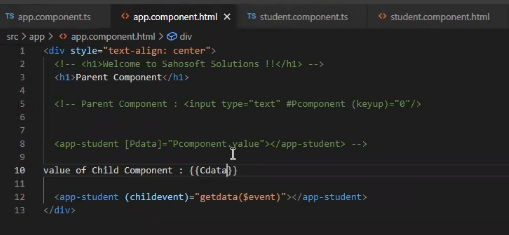
How to use Input

1. Data in html first



In ts now

Parent html



Why we use $ sign

For value taking from event

Providers Property

Providers we use for services.

Major change in angular 6.

Module level and component level service difference is same as class.