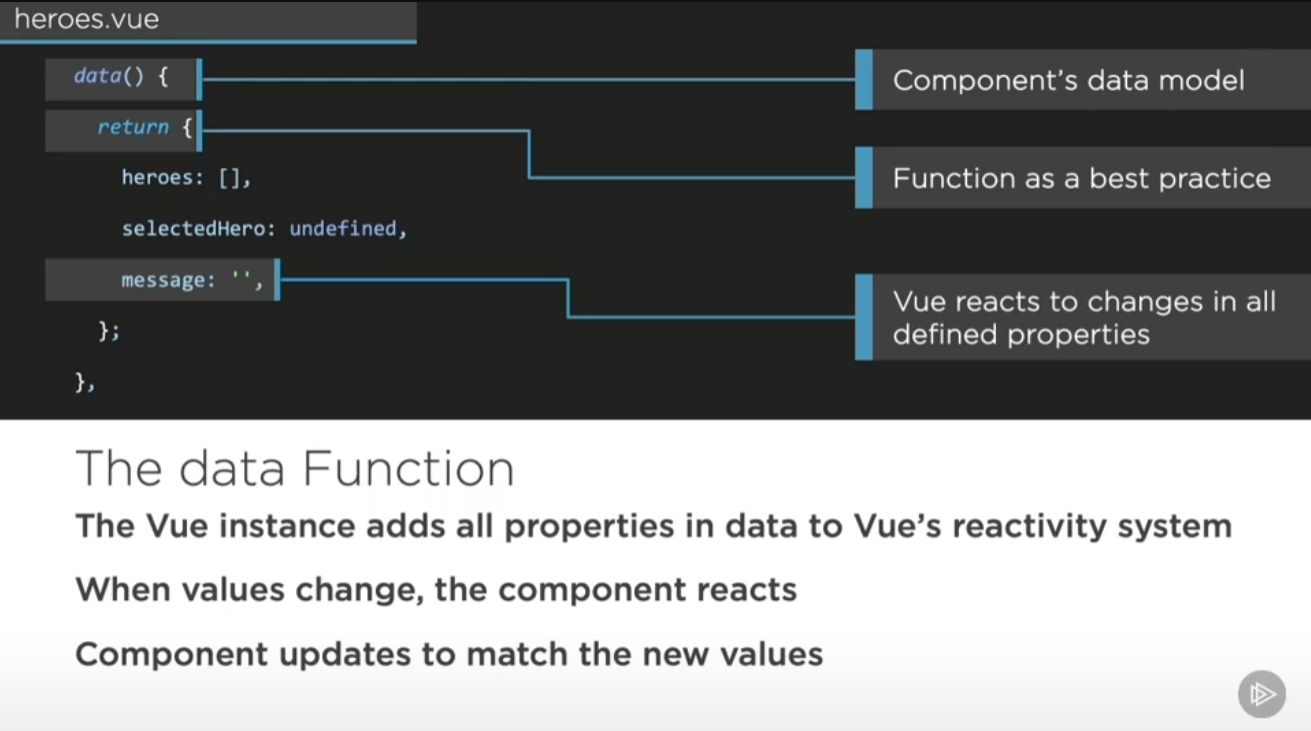
**L2.5: Component**

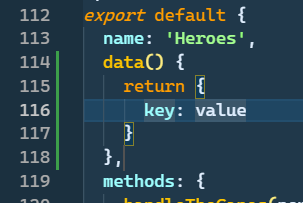




**>> event thought initial state is undefined, still have to define this var for vue to keep an eye on this property**

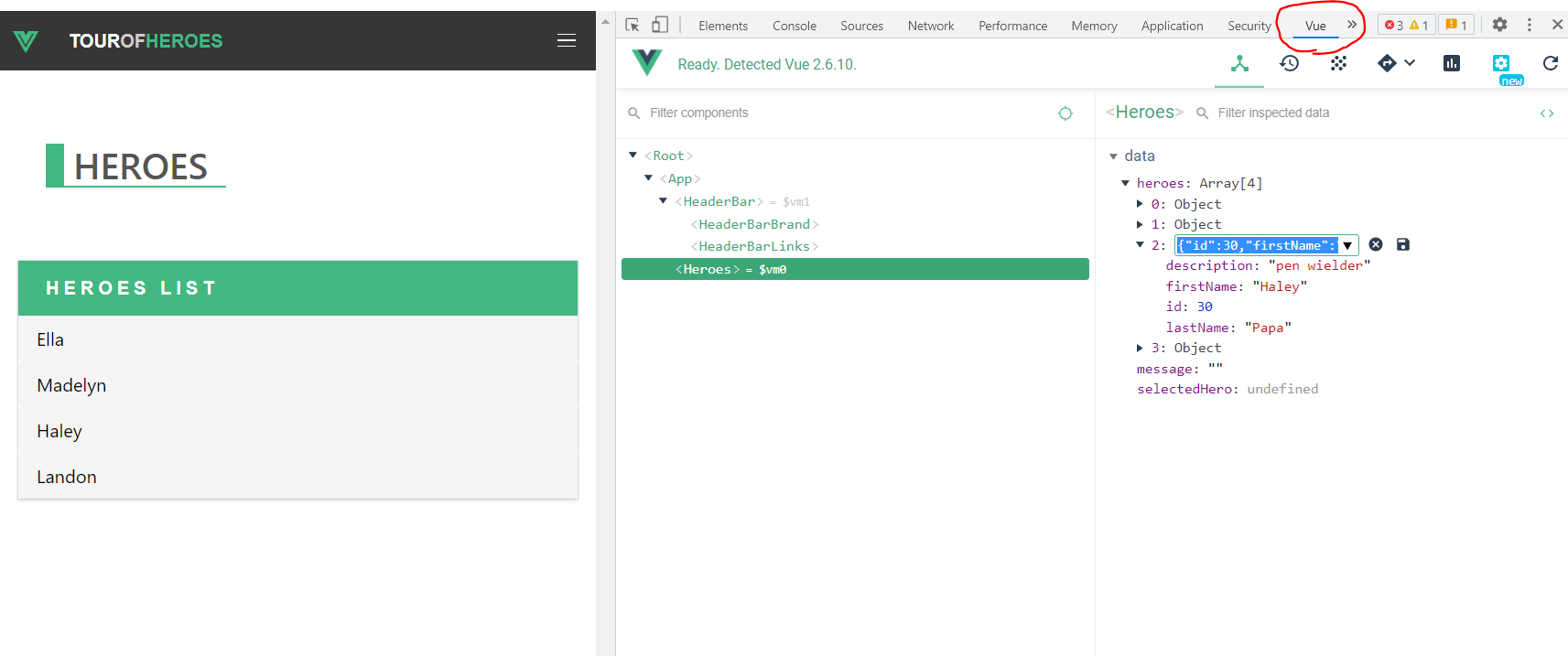
**>> when start, vue first see the properties inside this model function, then it know what properties(model) to focus on.**

1. **Type “vdata” and then tab to quickly generate data function (from vue snippets extension)**

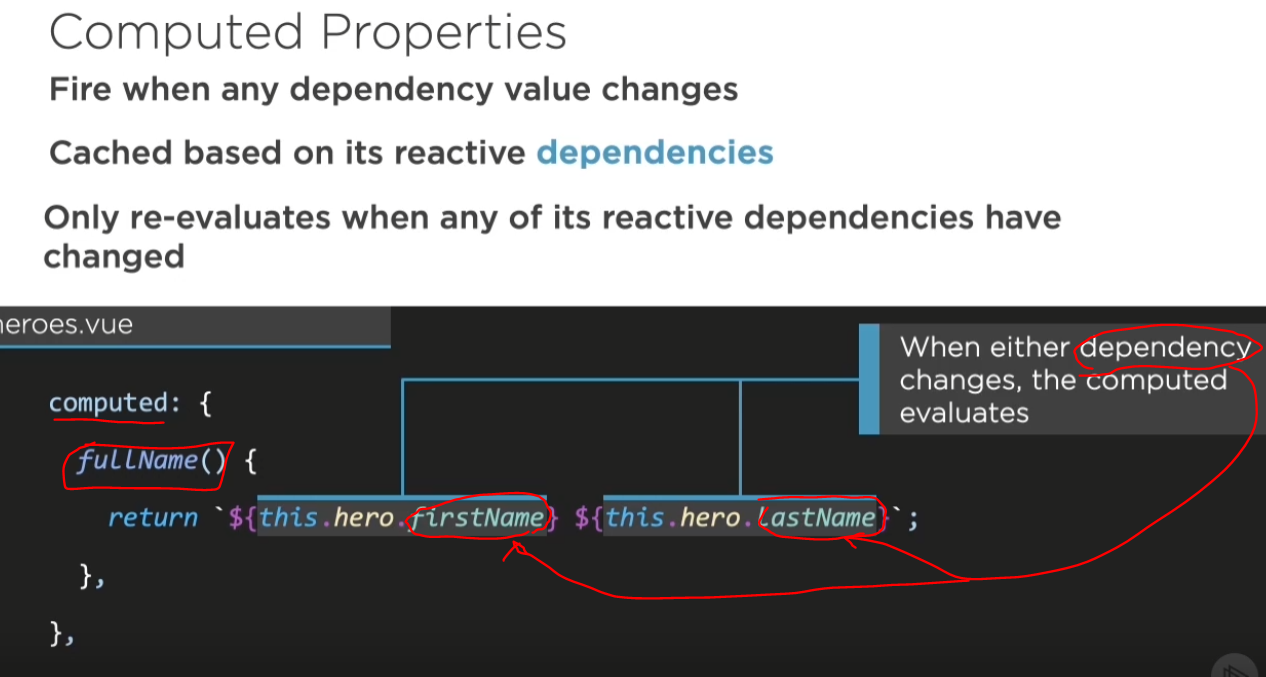


1. **Use chrome extension on vue let you edit and modify your modle inside browser**

**(just like style editing)**

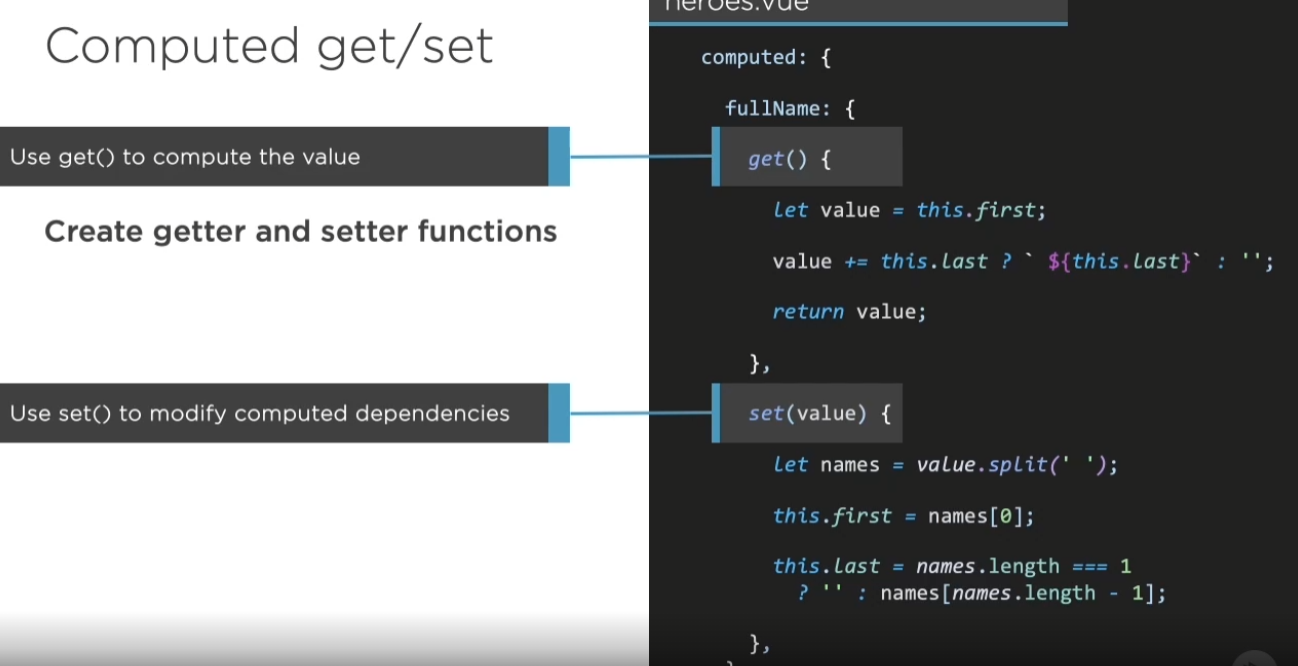


* **Computed properties (property depends on other properties)**

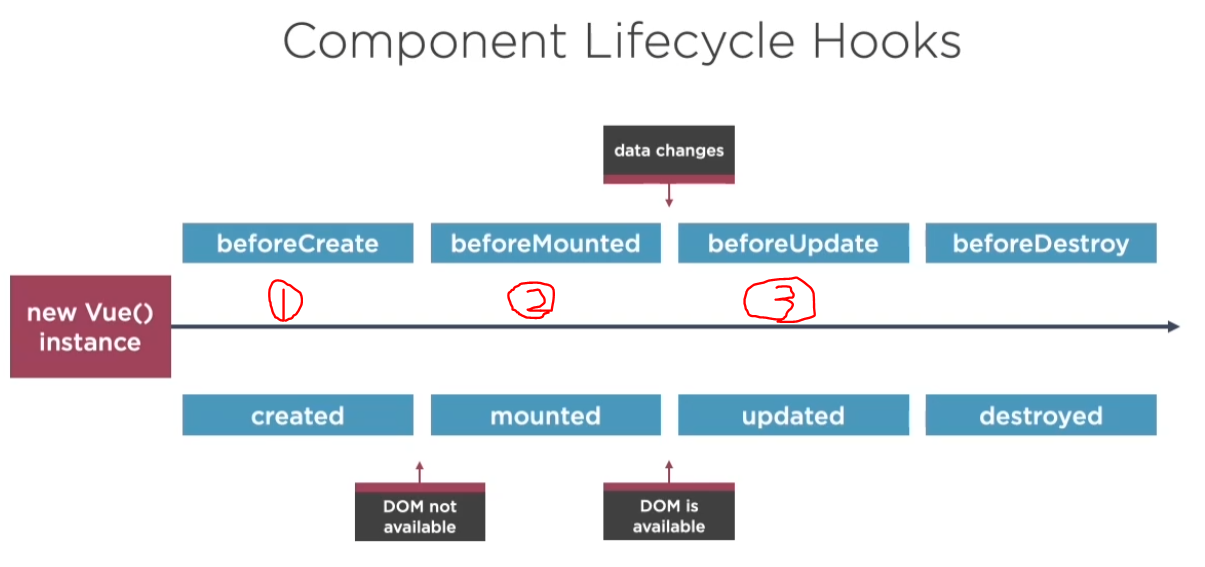


**>> or better use getter and setter**

**Same way use snippets >> vcomputed**



* **Life cycle of a component**



1. **Created state:**

**>> DOM is not mounted yet, best timepoint to get your data from api or other backend**

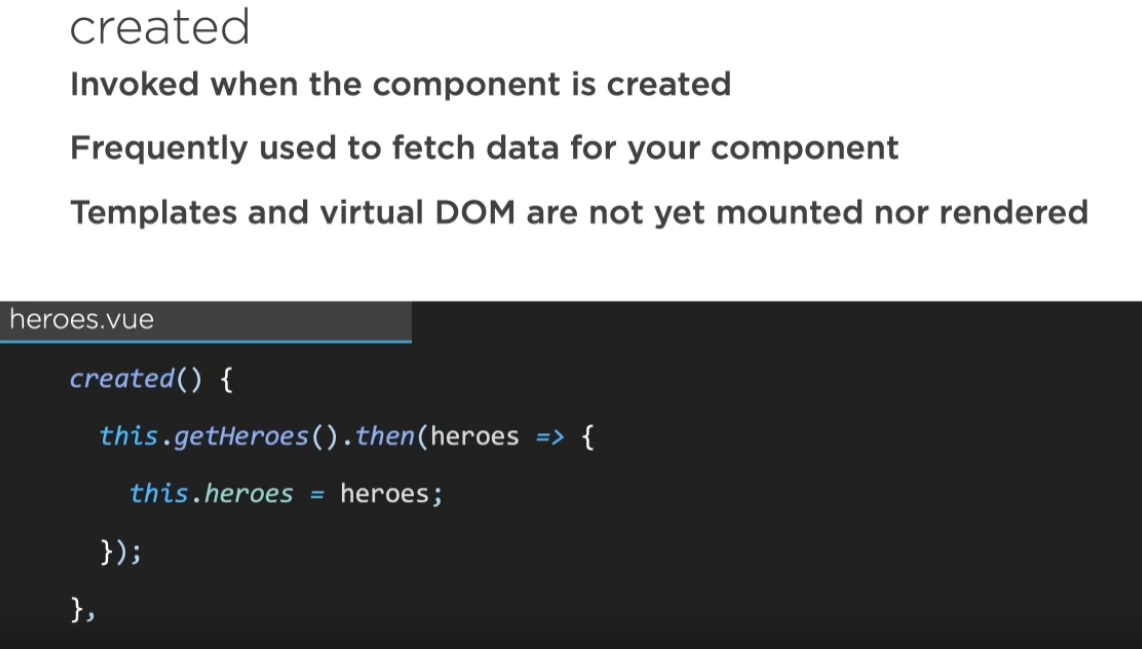
**>> 像angularjs 的vm.$oninit**

1. **Mounted state:**

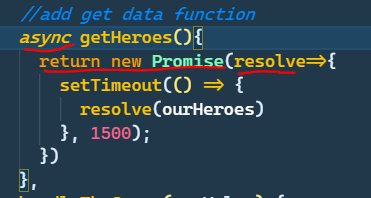
**>> time point to interact with other third party component (jquery or angular js etc.)**

1. **Model changes**

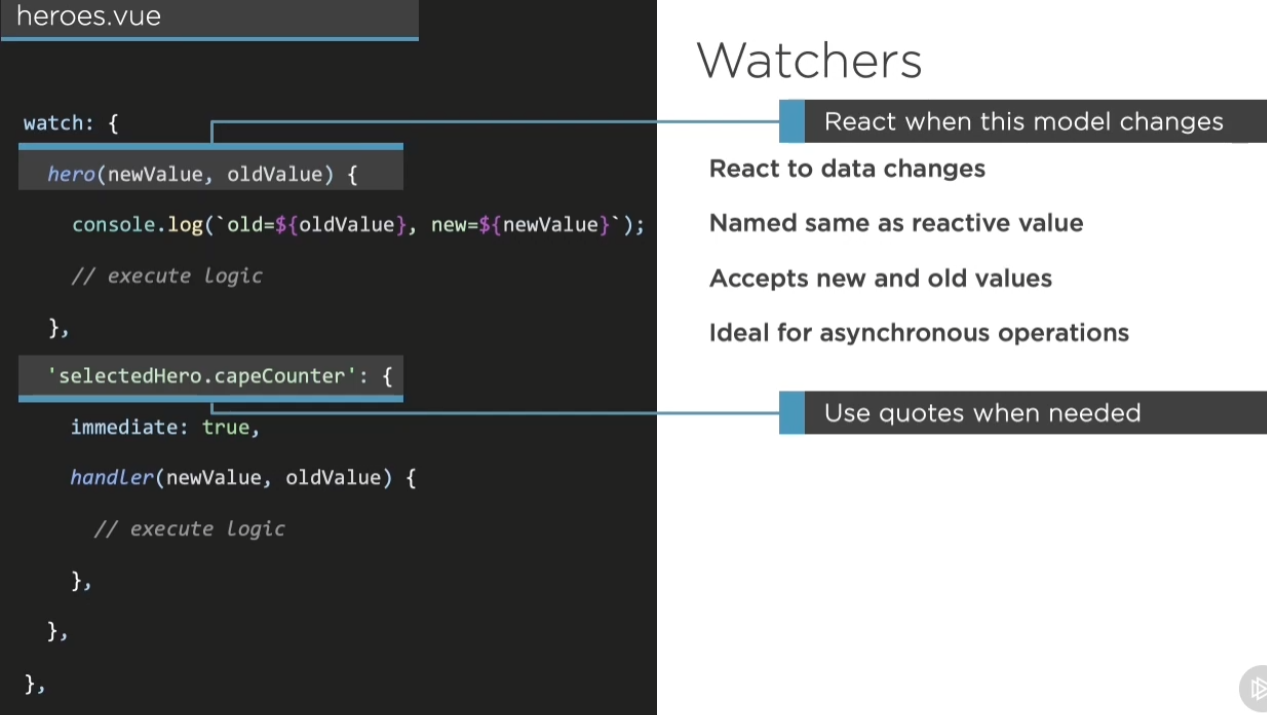


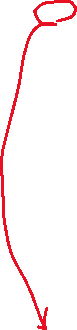
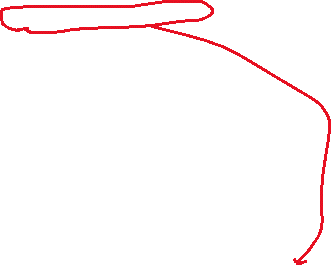


**\*\* way to create async function with promise**



* **Beside reactivity system, you can watch on a property actively using watcher.**





**Here hero is the property we try to watch**

**Here we watch a property inside defined property**

**\*(remember if it’s a dot property you have to put it in quote)**

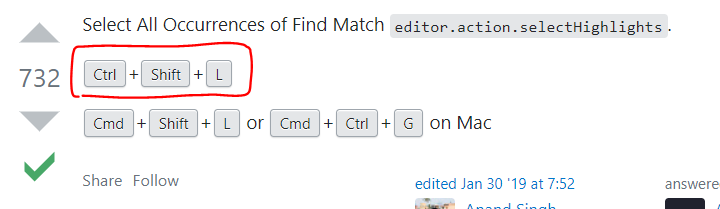
**>> set immediate to true means it will run the watcher immediately whether the property been changed or not. And set it to true will also need a handler**

**>> use vwatcher or vwatcher-option(full set version)**

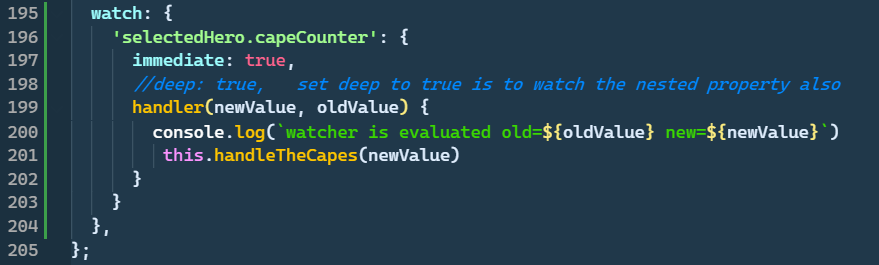
**\*\*\*好用技巧!! >> select all occurrences**

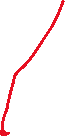
**>> 當要快速修改或者加行時可用**

**>> 先選取目標文字，然後 ctrl + shift + l**



* **Create and run a watcher demo**





**Whenever the capeCounter property is changed > things inside the handler will be evoke**

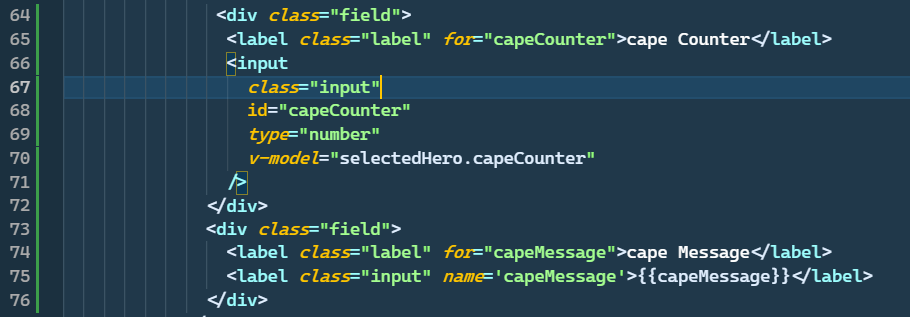
**>> here the handleTheCapes is the function defined in methods**

**If deep set to true, then things changed inside .capeCounter will also be watched**

**>> if changed the function will be evoked**

**(html demo also)**

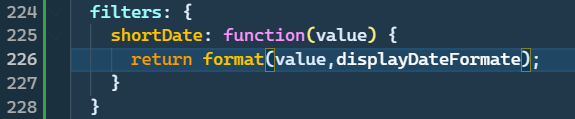
**>> the input here will change the capeCounter property and hit by watcher**



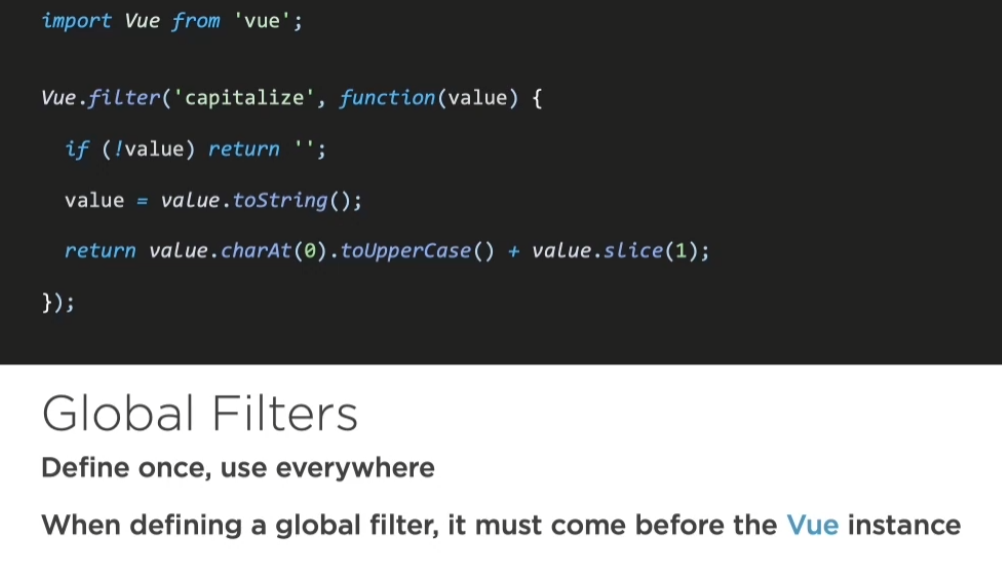
* **Create custom filter >> also defined in your component**



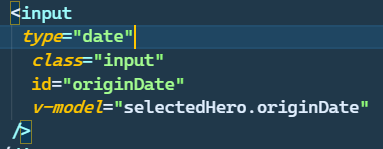
**Demo:**



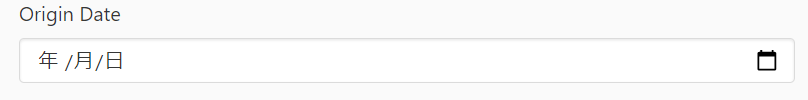
**>> or define filter globally**

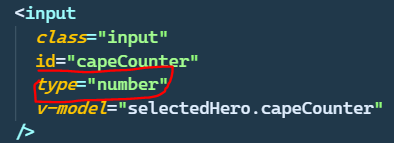


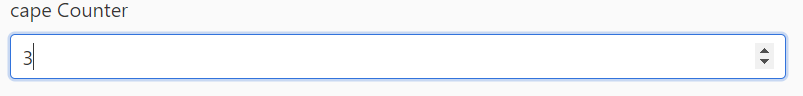
**Html筆記: input 的type property (看搭配的scc)**











**\*\*小技巧: 開新的terminal，同時舊的可以繼續執行**

**>> 就的繼續開著server，新的拿來npm install**



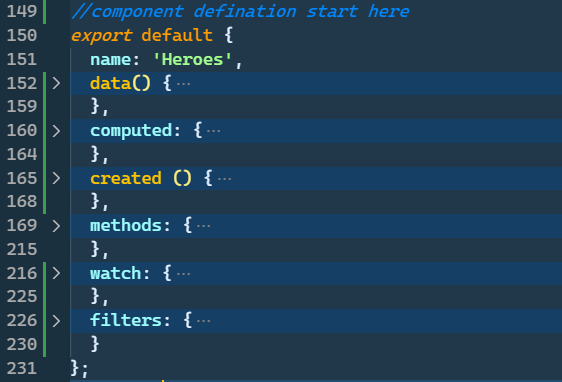
**>>** 



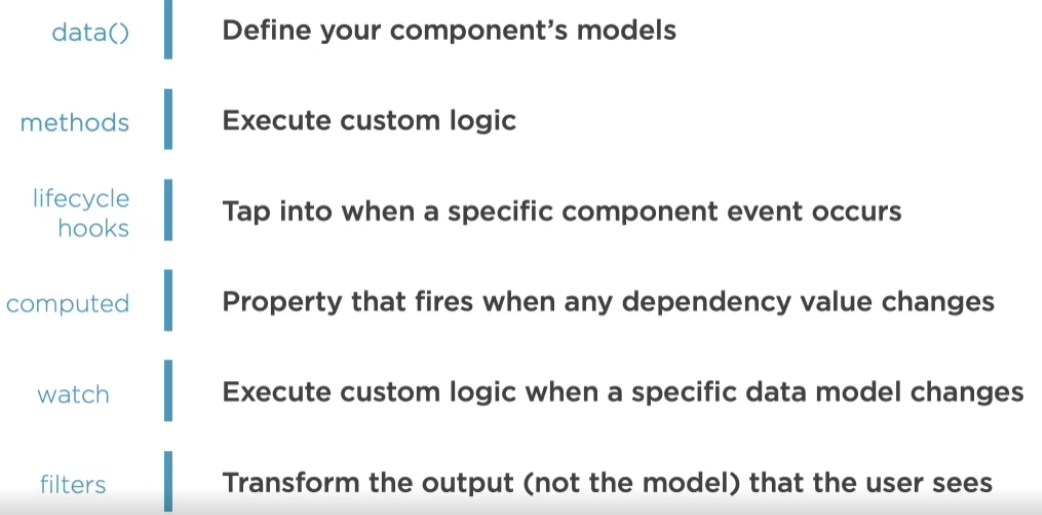
**>> node 的 date function**



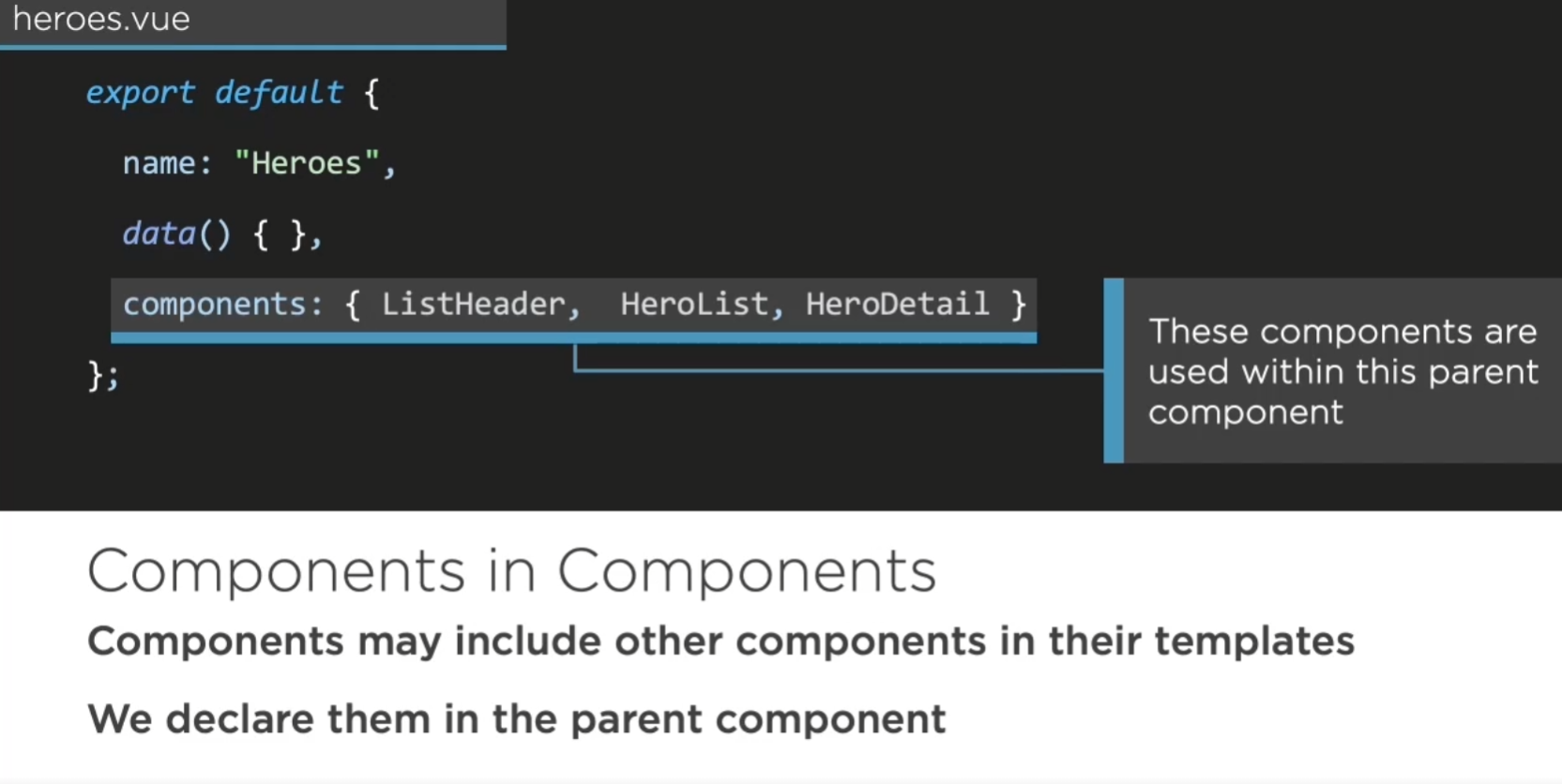
**Recap: component overview:**

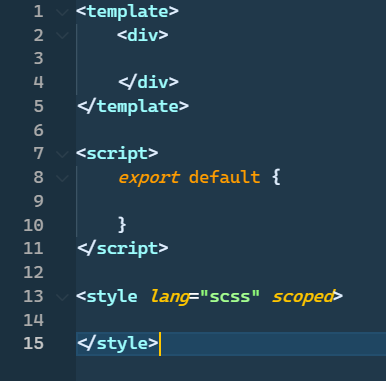




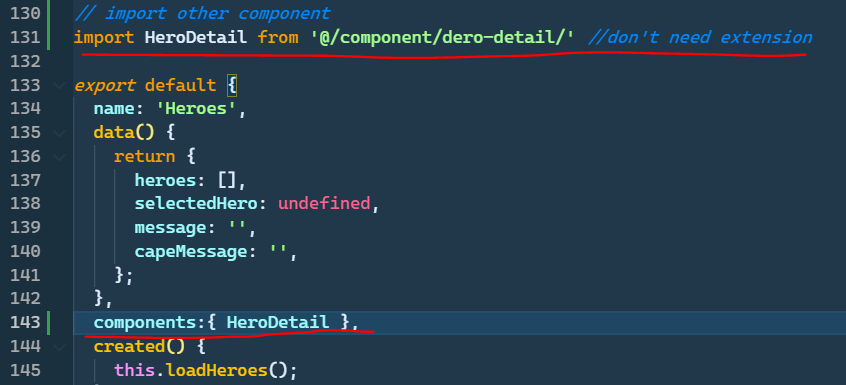


* **Component communication**

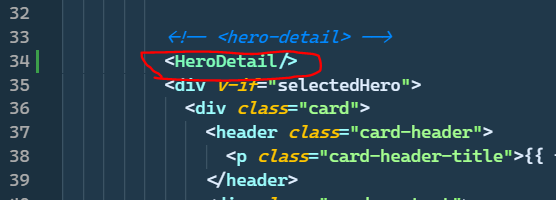
1. Component inside component 
2. 開一個子component的vue檔，然後利用vbase snippet快速建立底板

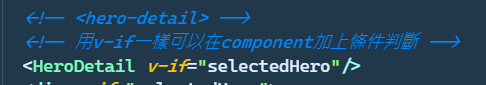


1. 建好之後去母component先import，再在下方建components

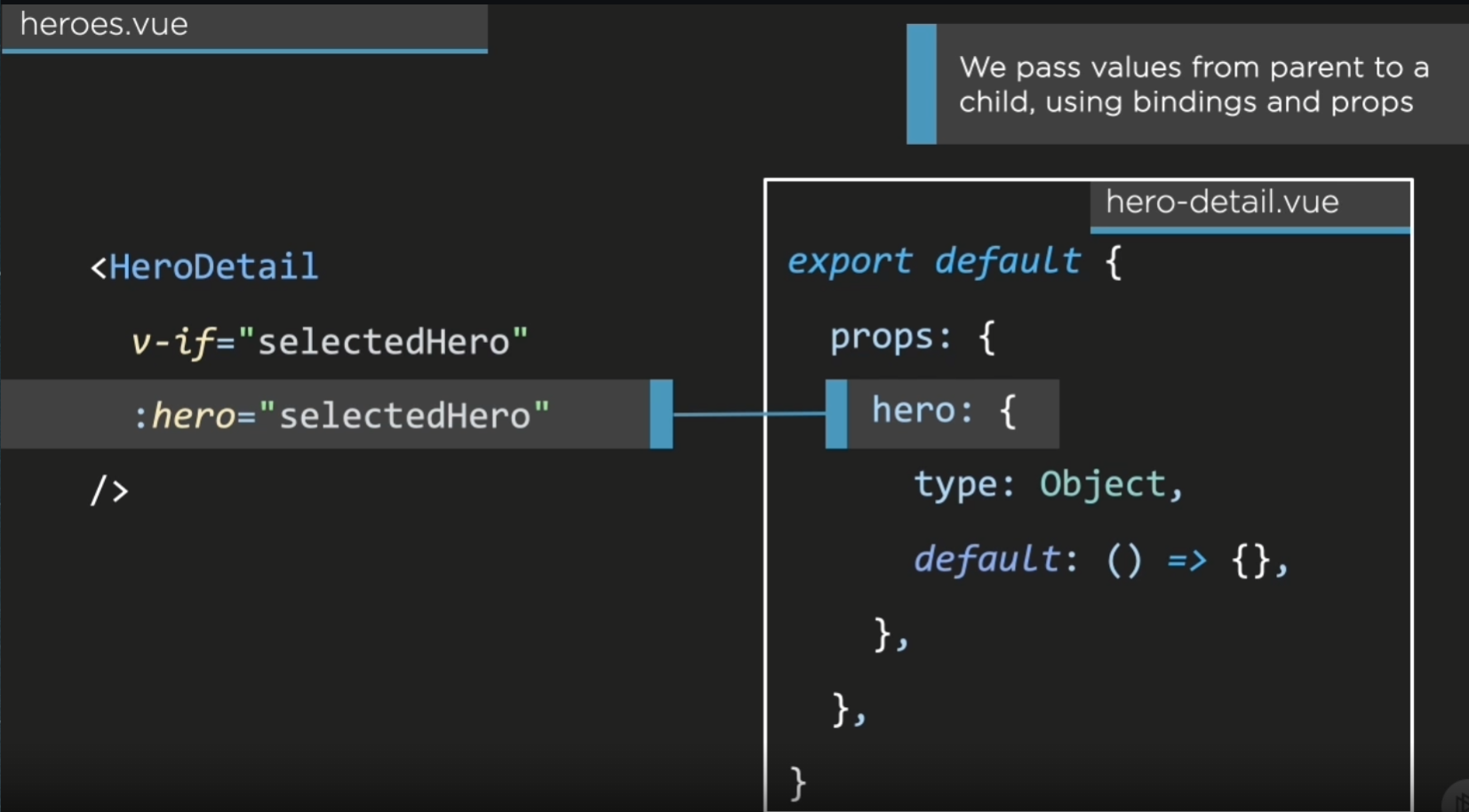


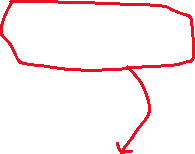
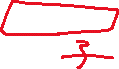
1. Finally put your component inside your template





* Pass value from parent to child using “props”

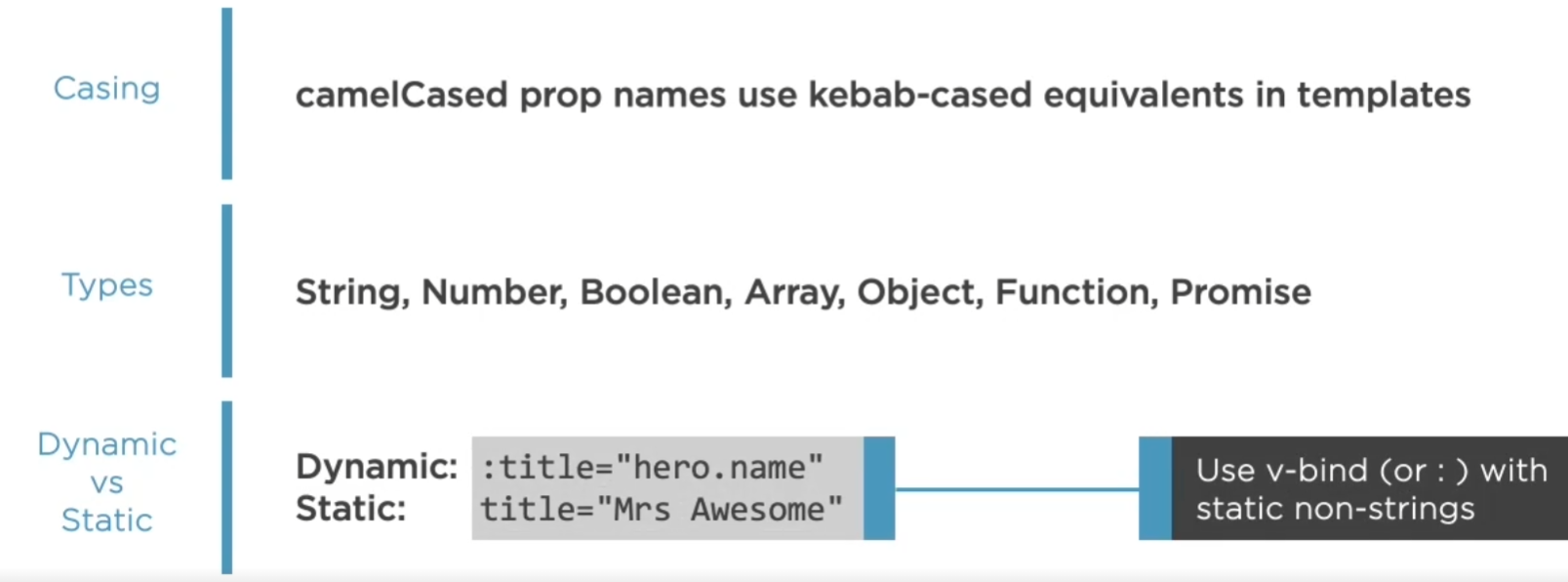




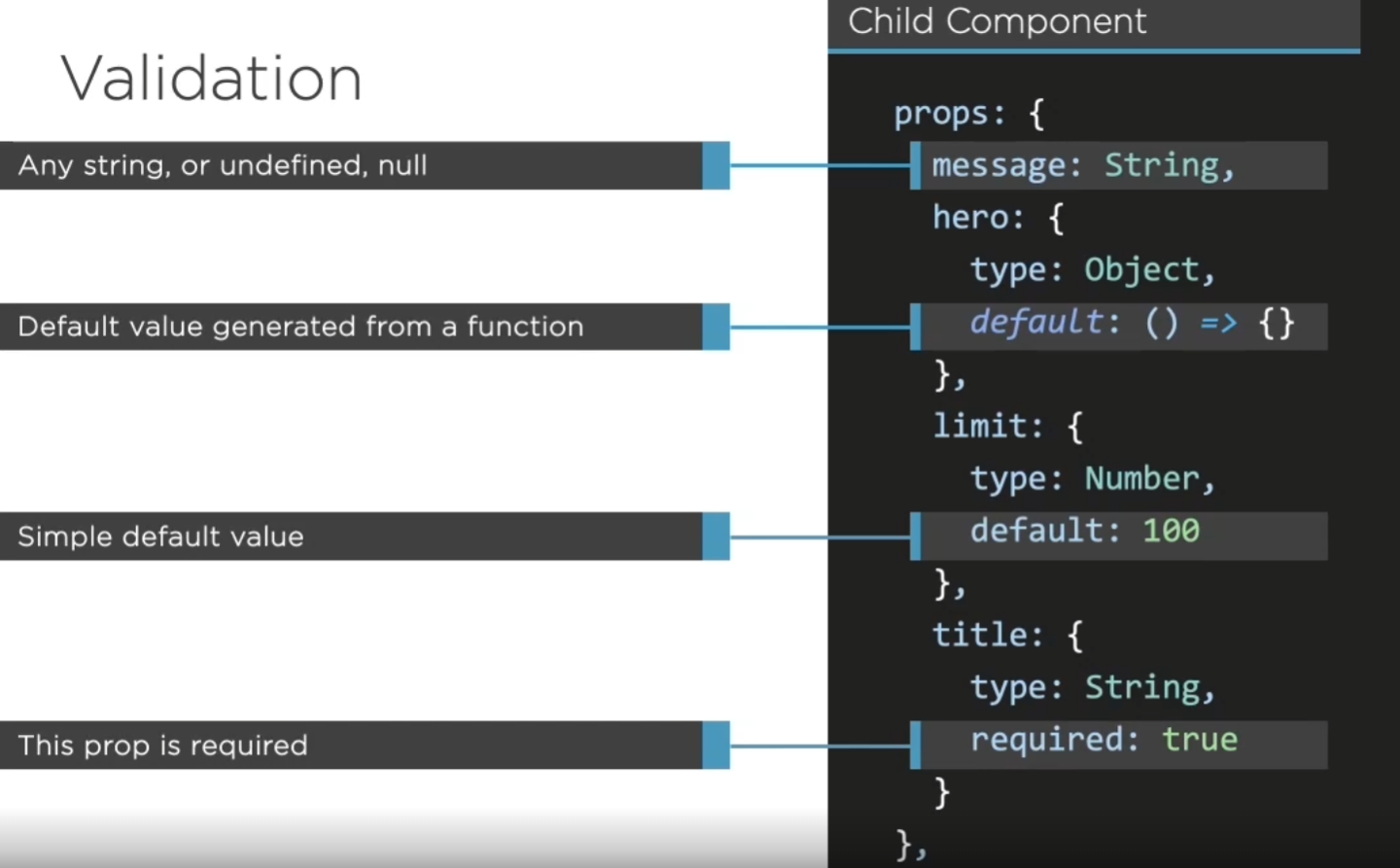
>> 在子component中先定義出props的型別(確保接過來時不會錯) >> default是空物件

>> 從母component中直接v-bind data進入子component中

注意事項:

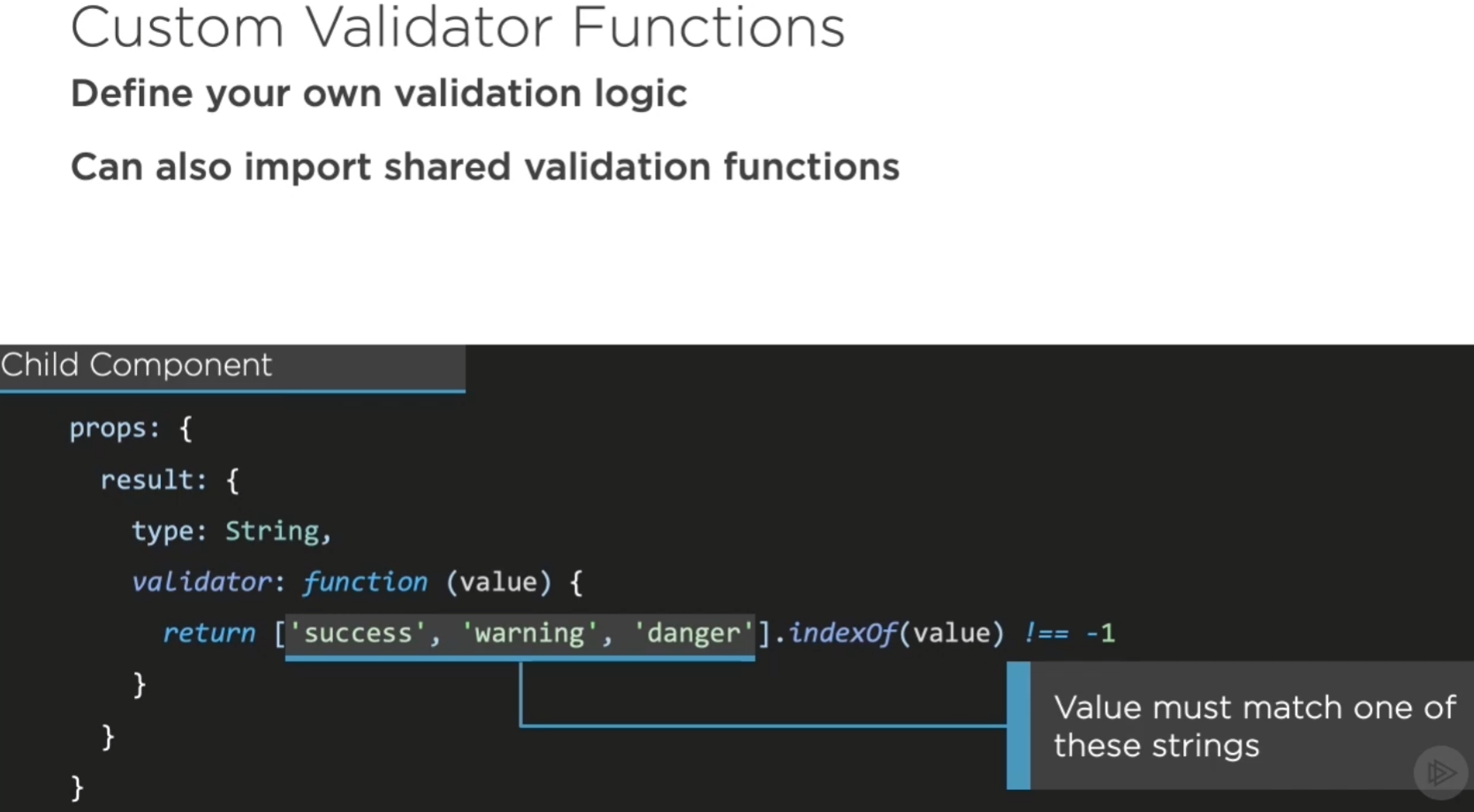


>> lowercase with dash: “kebab-case”





Customized validator:

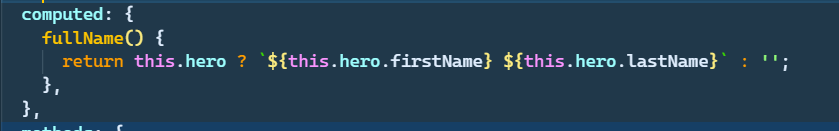




>> 使用arr.indexOf(value)可以快速判斷一值是否在arr中，如果回傳 -1 代表不在，回傳數字則是該元素位置

* Take one component and split it into parent and child component (if com is getting too large)
* >> 就把code貼到子component就對了

Point:

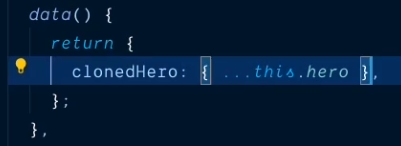


轉貼過來的code，如果是自component的prop的話記得加上this

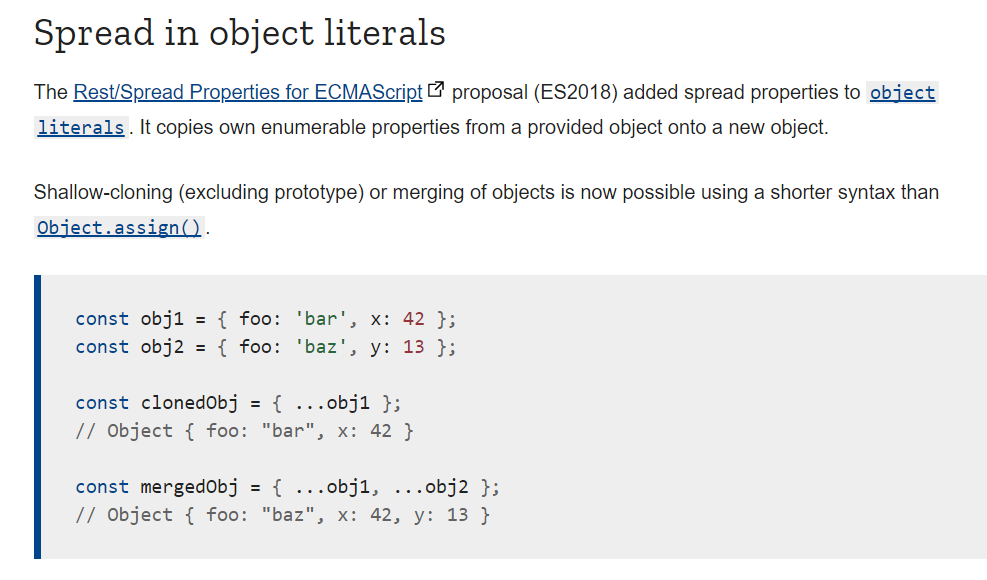
Q: how the object been passed?

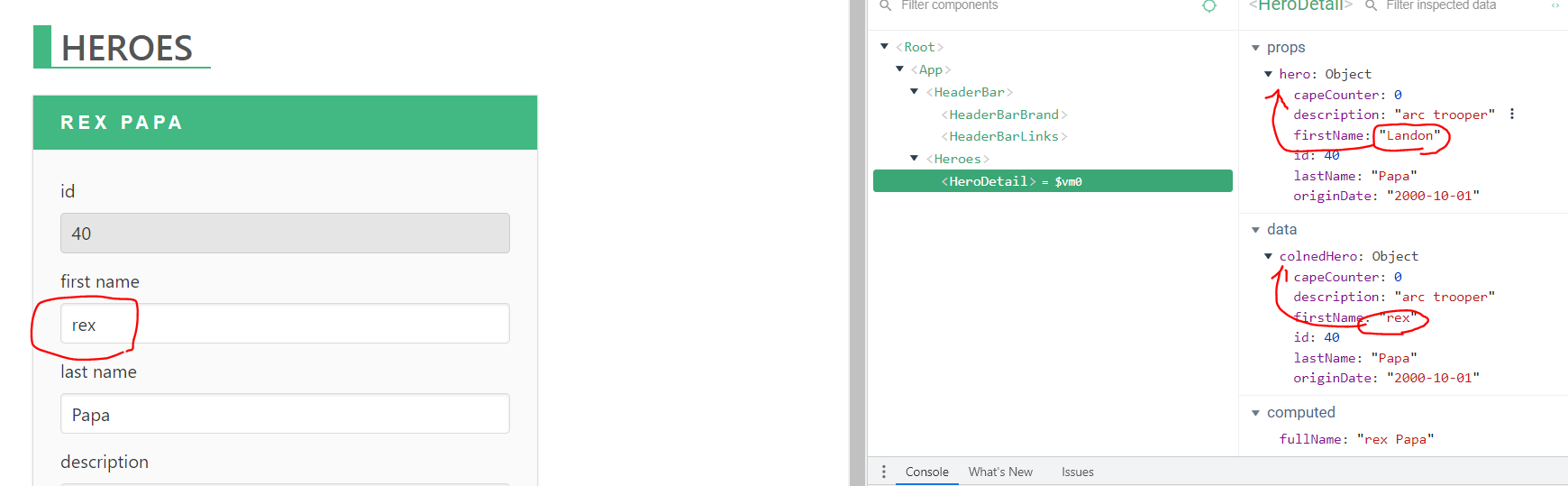
>> **If pass the object from parent to child through biding, and the child mutate directly, then the object in parent will also be mutated, and that is not we want. “We only want data to be thrown one direction, at least the direction we expected.”**

**>> so, when the child take the object from parent, first to do is to make a duplicate**



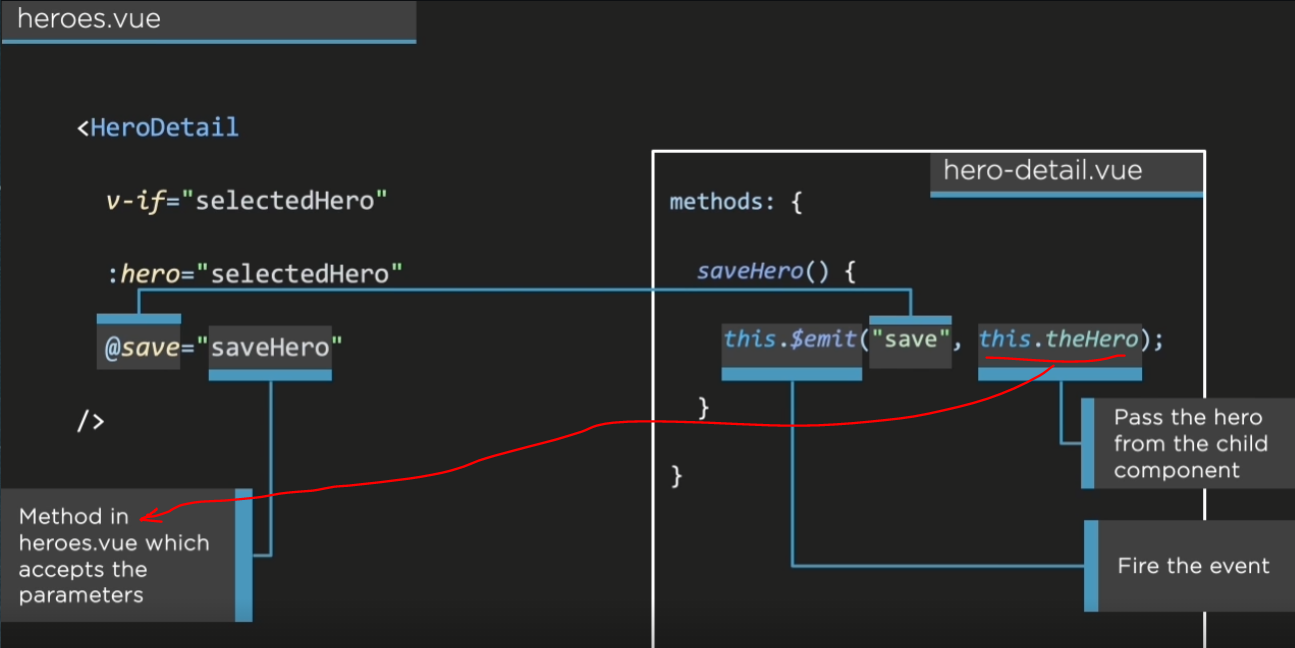
**Note: spread in object literals**



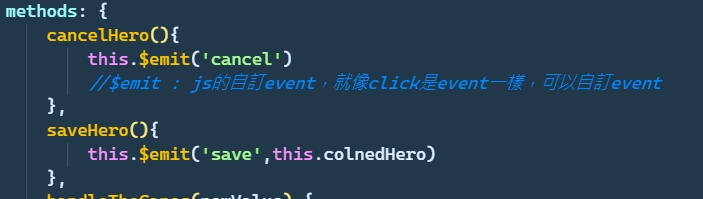


**>> now the origin data(parent) won’t be mutate by child**

* **Communicate from child to parent (method)**



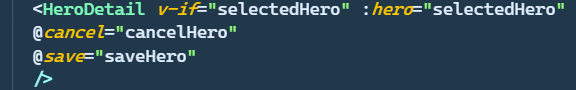
**子component:**

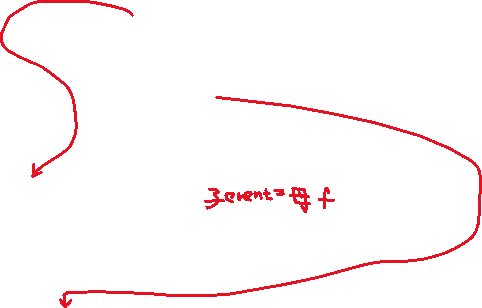


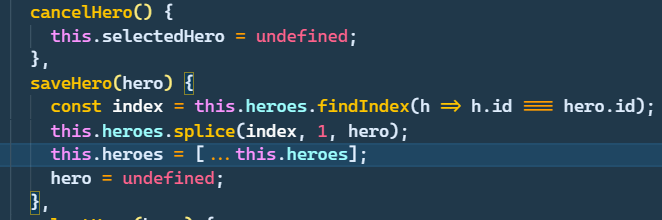


**母component**

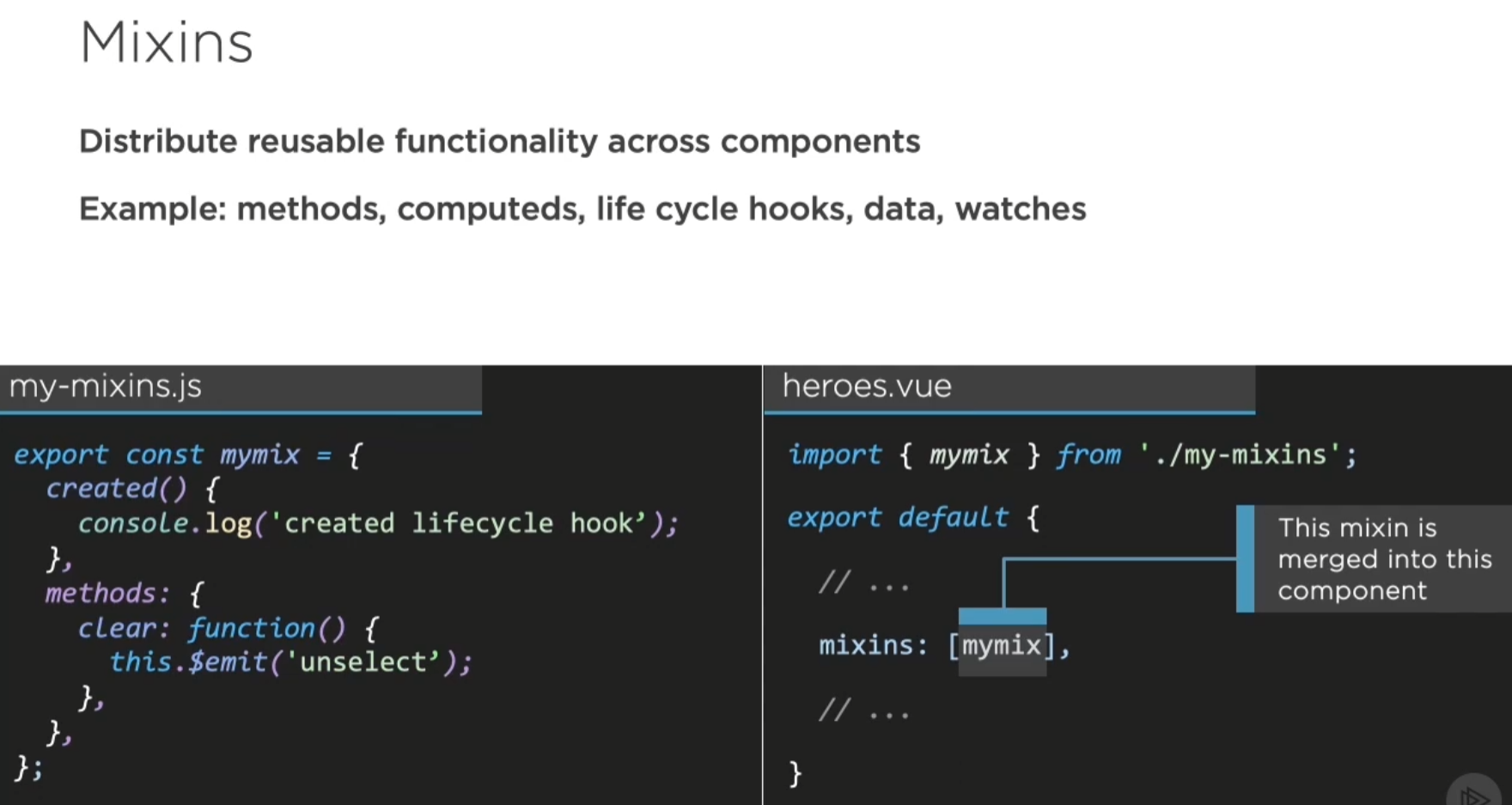




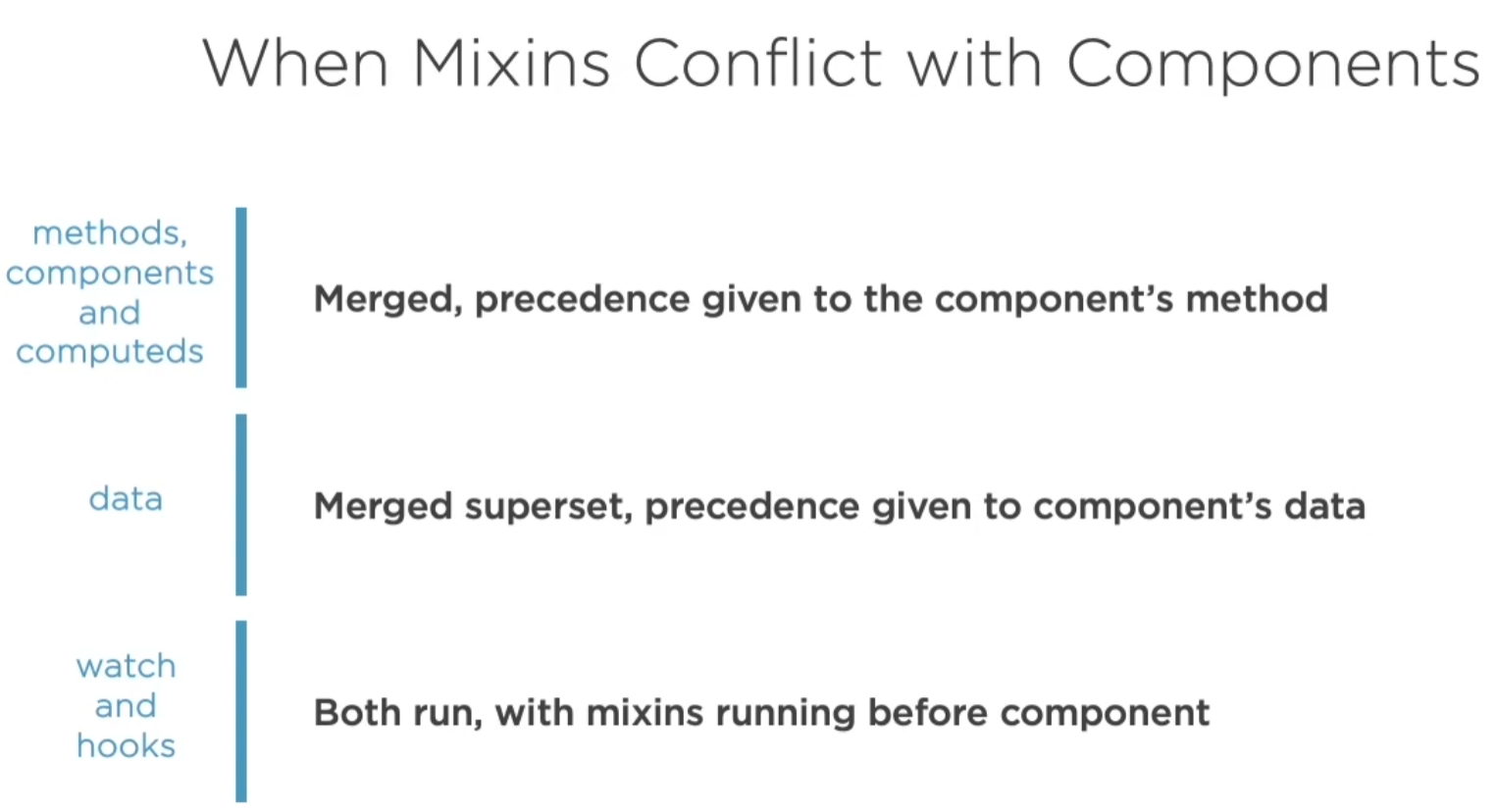




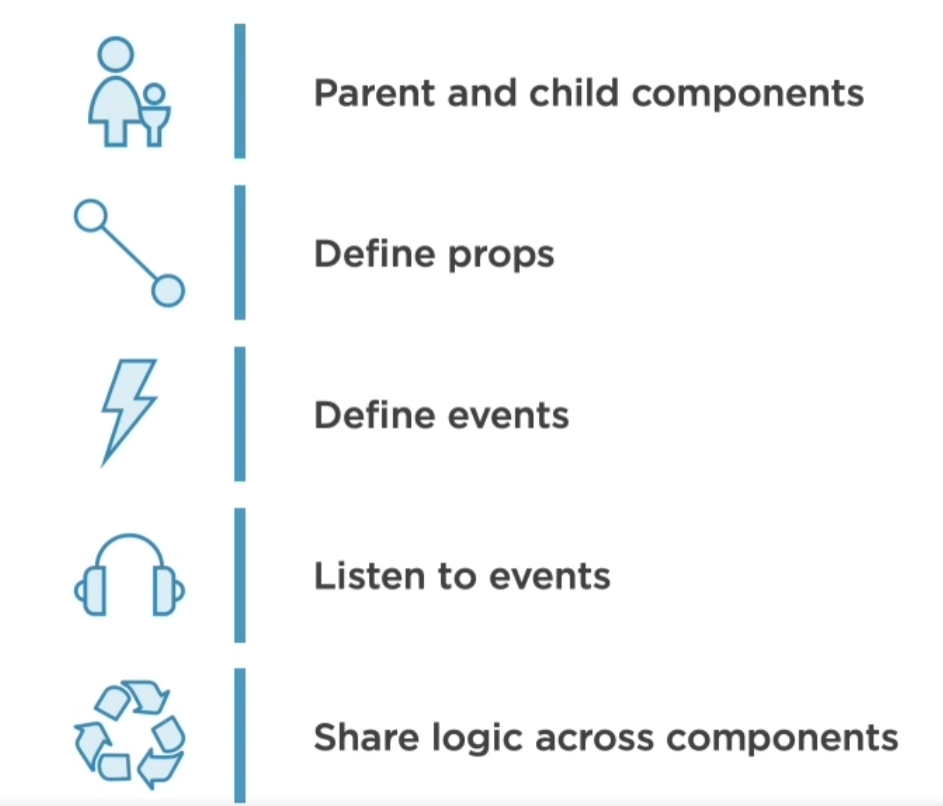
* **Share logic among components >> using mixins**



**What if the mixins you take in has the same name with local variable? (conflict)**



**Recap:**



**Summary:**

