<https://angular.tw/guide/glossary#lifecycle-hook>

life hook順序



**在觸發ngOnInit前會先觸發ngOnChanges**

**Angular bootstrap process :**

1. **組件實例化:初始化和注入(constructor,DI)**
2. **輸入屬性綁定 (@Input,ngOnChanges)**
3. **變更檢測**
4. **DOM初始化**
5. **ngOnInit**

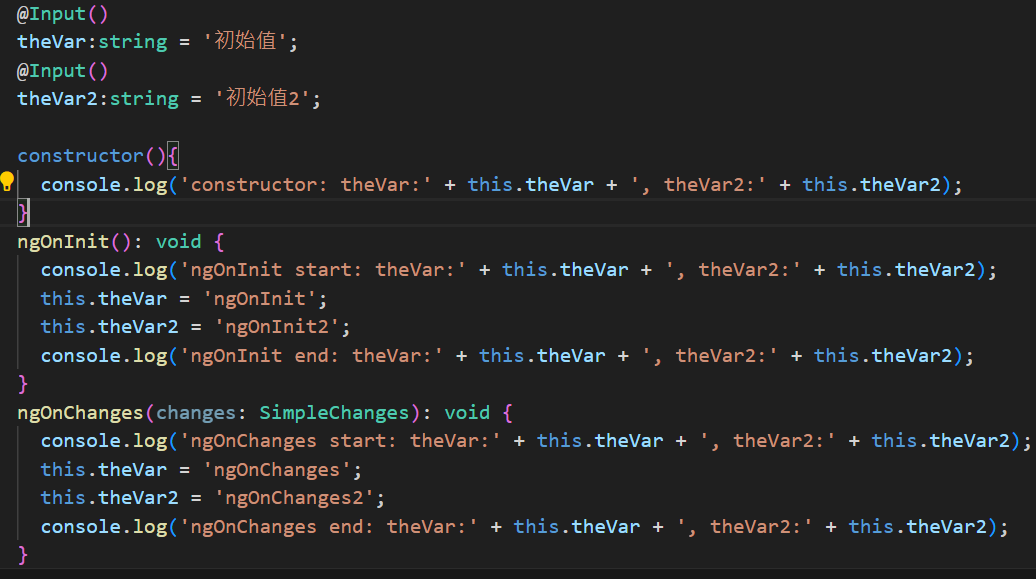
[@angular/core](https://angular.tw/api/core)

**OnChanges**

* a lifecycle hook,當指令的任一個可繫結屬性發生變化(為輸入屬性)時調用ngOnChanges()
* 小心效能因為容易一直觸發

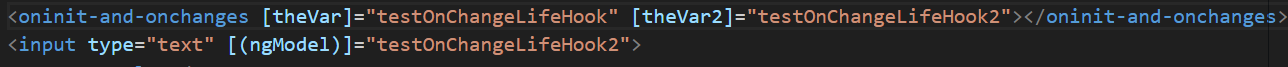
ex:

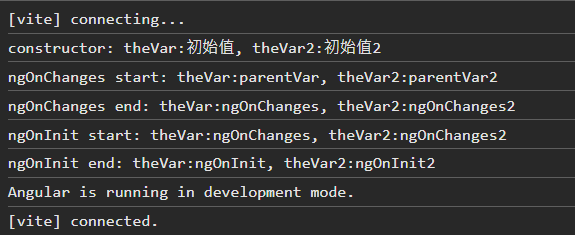
子元件



父元件



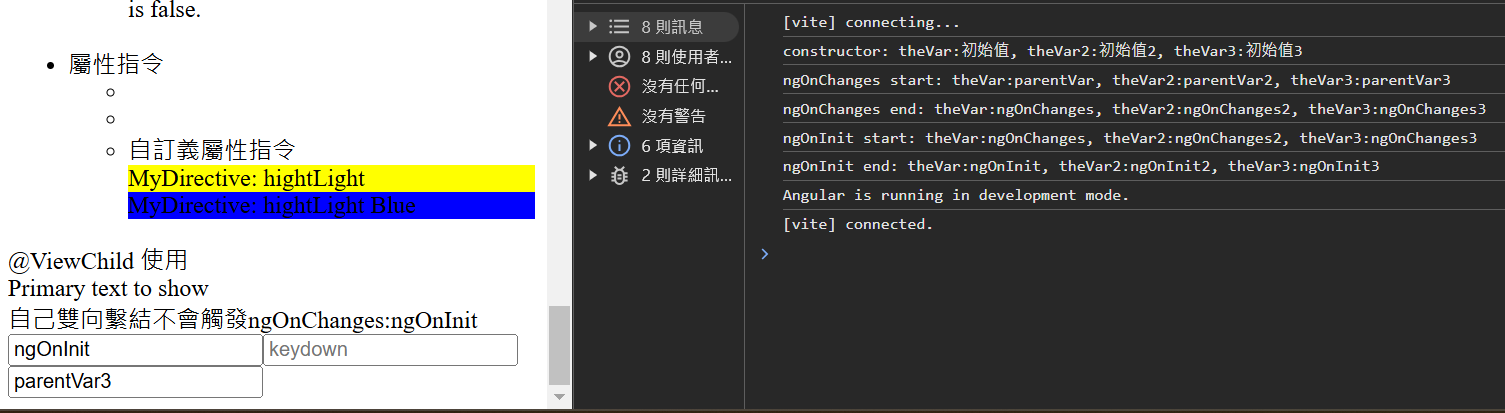


console.log畫面

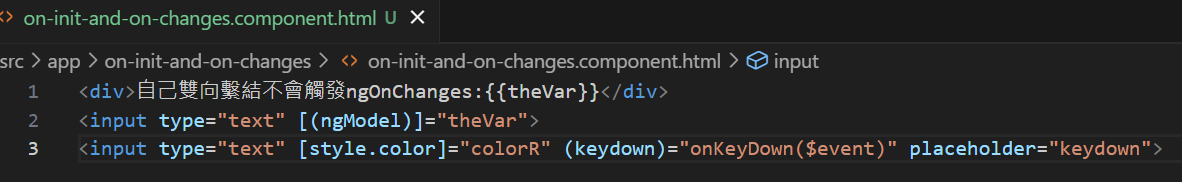
在初始化時ngOnChanges比ngOnInit早觸發

使用@Input時,自己的物件中改變不會觸發,父元件才觸發

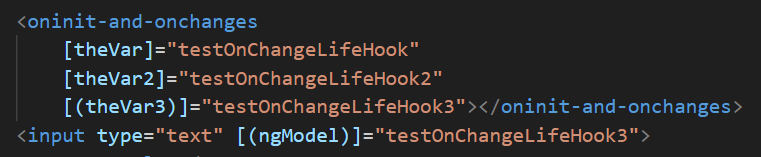
初始化後的輸入格如下



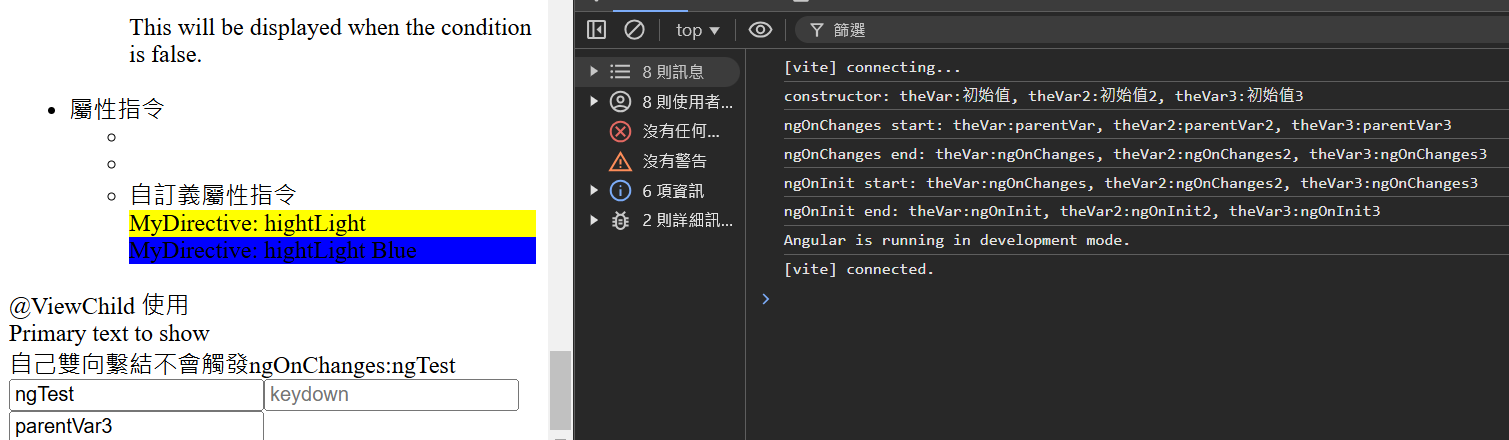
子元件的html



父元件html

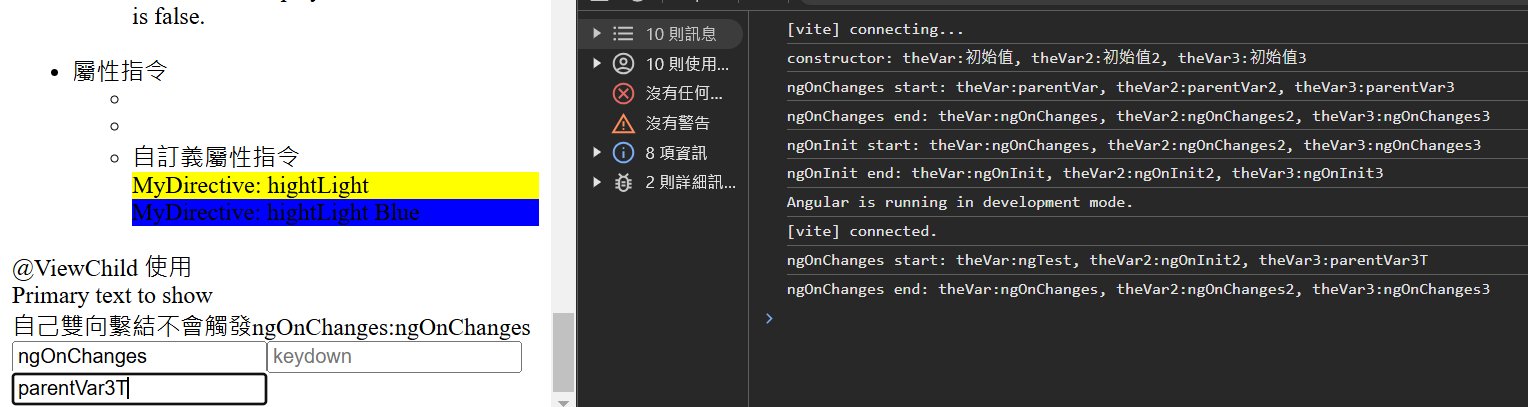


修改第一個input內容(子元件第二行)不會觸發ngOnChanges(..)



修改父元件input內容,會將資料傳遞給子元件,觸發ngOnChanges(..)

在察看結果時發現剛修改的theVar有被修改內容只是沒觸發ngOnChanges

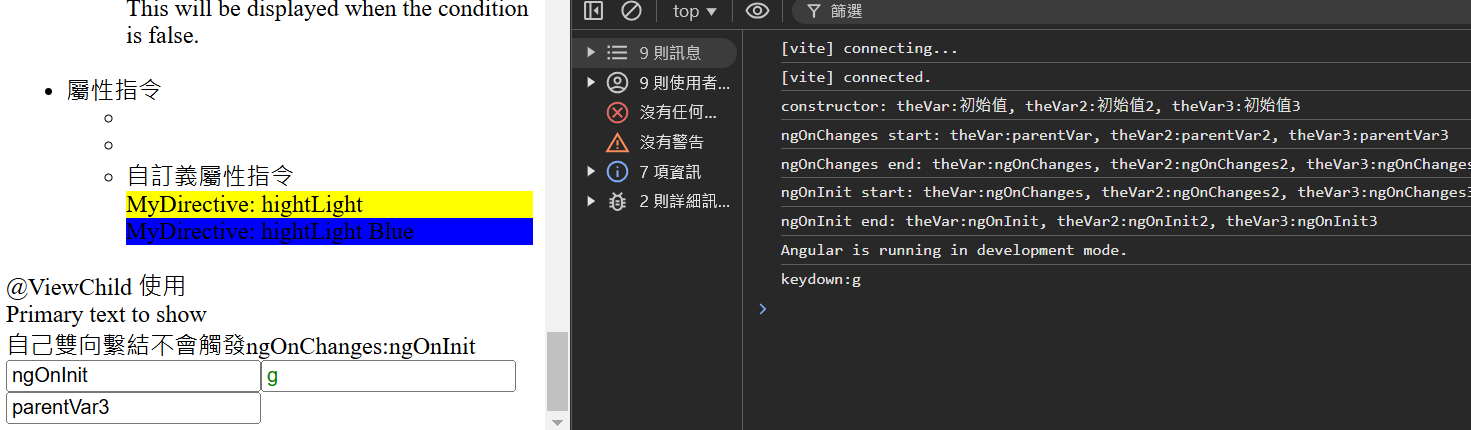


修改子元件第二個input(style.color)

子元件class內容



輸入值改變屬性時沒觸發



[@angular/core](https://angular.tw/api/core)

**OnInit**

* a lifecycle hook,在輸入屬性繫結後還有Onchanges後執行
* ngOnInit()只會執行一次
* 初始化的內容盡量寫在此

[@angular/core](https://angular.tw/api/core)

**DoCheck**

* a lifecycle hook,可自訂變更檢查 (每次angular進行檢查時呼叫)
* DoChanges無法監測到的部分可以用此,但比DoCheck更容易被觸發,code盡量輕量化避免影響效能
* 備註點擊input或切換輸入到不同input會觸發

[@angular/core](https://angular.tw/api/core)

**AfterContentInit**

* a lifecycle hook,第一次的DoCheck後執行一次

(即使沒有實作DoCheck還是會在被執行)

(在 Angular 完全實例化了指令的所有內容之後呼叫)

* 用在監控內容投影(content-projection)響應完成後的變更

(父元件傳遞給子元件的內容投影完成時觸發,僅一次)

[@angular/core](https://angular.tw/api/core)

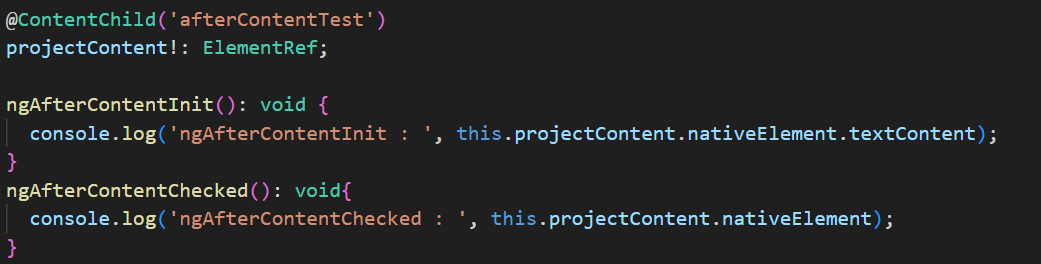
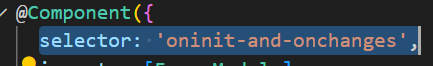
**AfterContentChecked**

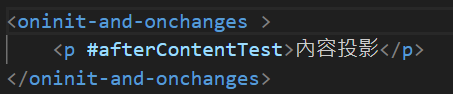
* a lifecycle hook,在每次DoCheck後都會呼叫
* 每次變更檢測時,檢查投影時觸發(AfterContentInit和DoCheck後執行)

ex:

子元件



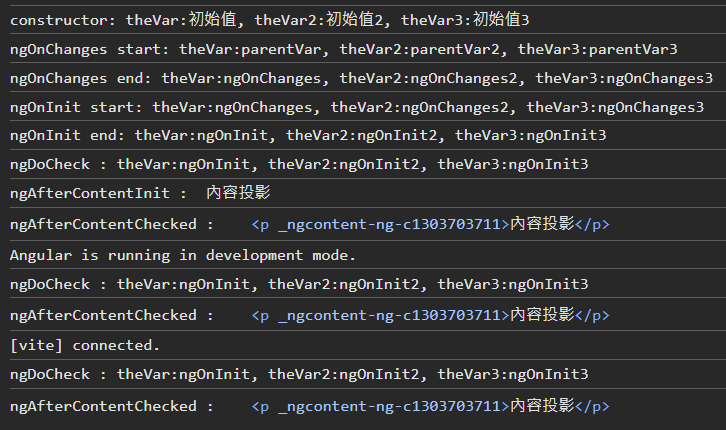




console

只會有一次AfterContentInit,

每次檢測投影都會跑一次AfterContentChecked



[@angular/core](https://angular.tw/api/core)

**AfterViewInit**

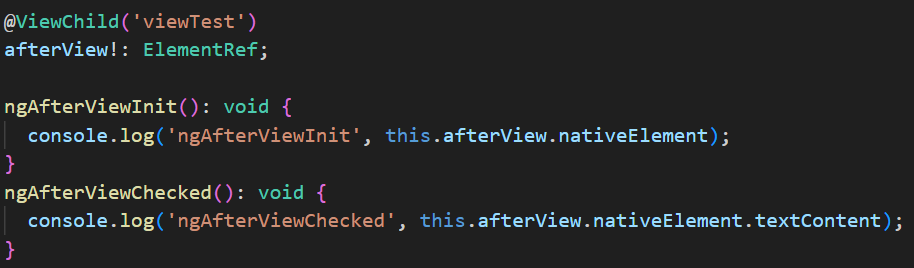
* A lifecycle hook,在完成元件view初始化後呼叫，僅呼叫一次
* 此時可以操作DOM(因為已完成view)

[@angular/core](https://angular.tw/api/core)

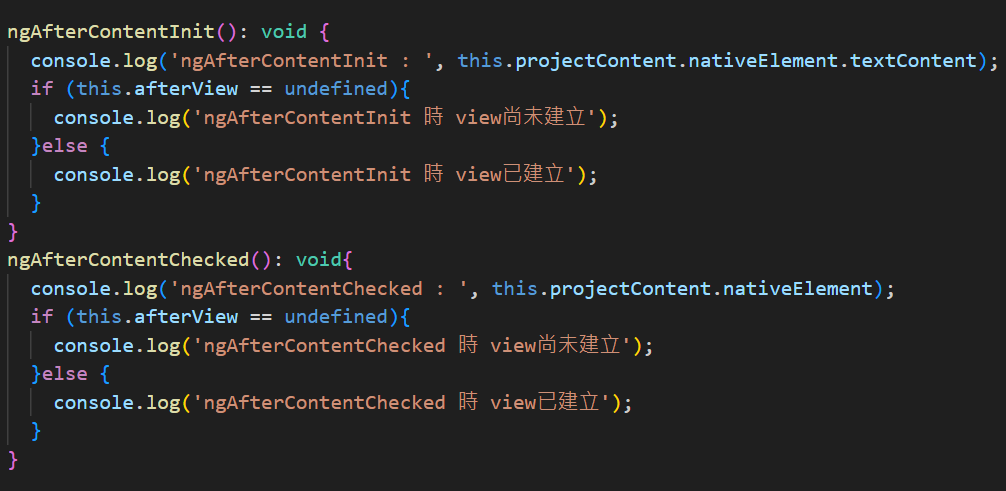
**AfterViewChecked**

* A lifecycle hook,在每次AfterContentChecked後呼叫(即使沒有也會呼叫)
* 在AfterViewInit後也會被呼叫
* 在@Input屬性變化時也會被呼叫
* 監控View變化

ex:子元件



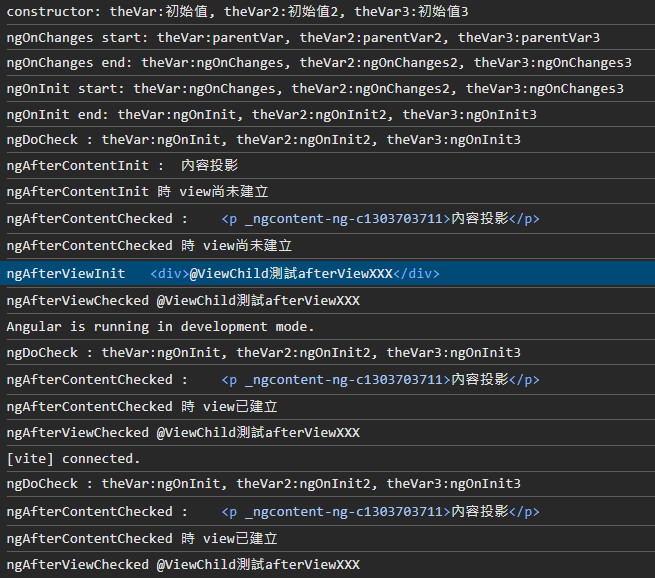
子元件Content時加入view確認



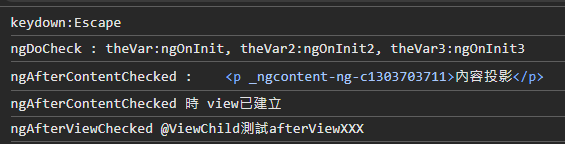
子元件html



從打印可看出view在ContentInit時尚未建立



input被聚焦或改變內容時會一起被觸發



[@angular/core](https://angular.tw/api/core)

**OnDestroy**

* A lifecycle hook,當指令(directive)或通道(pipe)或服務(service)被銷毀時調用
* 常用於:路由導航離開當前組件前、父組件移除子組件時、手動銷毀動態組件